Universidad Autónoma de Baja California
Facultad de Ingeniería

Interfaz para la interpretación del movimiento ocular a partir del procesamiento de señales de EOG.

Tesis

para obtener el título de

Maestría en Ingeniería

presenta

Irving Iran Reyes Nogales

Directora de tesis: Dra. Wendy Flores Fuentes
Codirectora de tesis: Dra. Monica Valenzuela Delgado

Mexicali, B.C., México Septiembre de 2025

Resumen

Las señales obtenidas mediante un EOG normalmente son utilizadas para detectar patologías (trastornos oculares y neurológicos) en pacientes como la toxicidad en la retina por consumo de ciertos medicamentos o enfermedades relacionadas con la alteración del sueño; sin embargo, las señales de EOG pueden ser interpretadas por un procesador con el objetivo de controlar diversos dispositivos (silla de ruedas, control de cursor en un computador, escritura) a través de una interfaz humano-computadora o simplemente para analizar el movimiento ocular. Las personas con discapacidad motriz pueden ser beneficiadas por la interpretación de señales de EOG dado que con los movimientos oculares se podrían controlar objetos o hacer determinadas tareas en distintos entornos donde se implemente el uso de estas señales. A lo largo de los años se ha intentado adquirir dichas señales e interpretarlas para posteriormente darles algún uso. El punto de interés de esta investigación es la interpretación de las señales, pero uno de los problemas que se presenta es que en cada sujeto de prueba las señales varían, ya sea por la colocación de los electrodos o por la fisiología del mismo paciente. Actualmente, en la interpretación de señales EOG, no se había obtenido un patrón estándar para utilizarlo en diversas aplicaciones sin necesidad de ajustes manuales hacia un individuo específico. En esta tesis se presenta el desarrollo de una interfaz para la interpretación de movimiento ocular horizontal a partir del procesamiento de señales EOG. Se adquiere en tiempo real la señal en DC filtrada con un arreglo RC que produce una señal en AC, la cual es caracterizada de acuerdo al usuario en un proceso llamado "Calibración", para posteriormente, aplicar un algoritmo de reconstrucción de la señal AC a la forma de DC atenuando el problema de la deriva de la señal, para la cual su amplitud es directamente proporcional a los desplazamientos oculares realizados. La interfaz entrega

en aproximadamente tiempo real las posiciones del movimiento ocular en pixeles y/o voltaje, los cuales son proporcionales a las dimensiones del campo de visión del usuario que está realizando los movimientos oculares.

Abstract

The signals obtained through an EOG (Electrooculography) are typically used to detect pathologies in patients, such as retinal toxicity caused by the use of certain medications or sleep-related disorders. However, EOG signals can also be interpreted by a processor with the aim of controlling various devices (wheelchairs, computer cursor control, text writing) through a human-computer interface, or simply to analyze eye movements. Individuals with motor disabilities can benefit from the interpretation of EOG signals, as eye movements could be used to control objects or perform specific tasks in different environments where these signals are implemented.

Over the years, efforts have been made to acquire and interpret these signals for practical applications. The main focus of this research is the interpretation of the signals; however, one of the challenges is that the signals vary across test subjects, either due to electrode placement or the individual's physiology. Currently, in EOG signal interpretation, there has been no standardized pattern that can be used across different applications without requiring manual adjustments for each specific individual.

This thesis presents the development of an interface for the interpretation of horizontal eye movement based on EOG signal processing. The signal is acquired in real-time in DC, filtered with an RC array that produces an AC signal, which is then characterized for each user in a process known as "Calibration." Afterward, a signal reconstruction algorithm is applied to recover the AC signal into a DC-like form without drift issues, with its amplitude directly proportional to the eye movements performed. The interface provides real-time outputs of eye movement positions in pixels and/or voltage, which are proportional to the dimensions of the user's visual field during eye movement.

Dedicatoria

A mis familiares y doctores involucrados en el proyecto.

Agradecimientos

A la Universidad Autónoma de Baja California (UABC) y a la Secretaria de Ciencia Humanidades y Tecnología (SECIHTI).

Índice general

Índice de figuras

Índice de tablas

Introducción

La mayor parte de las señales que aparecen en los ámbitos de la ciencia y la ingeniería son de naturaleza analógica, es decir, las señales son funciones de una variable continua, como el tiempo o el espacio y normalmente toman valores en un rango continuo [1, 2]. Para implementar el procesamiento de esas señales con el uso de sistemas de cómputo es necesario aplicar procesamiento digital de señales [3]. Un tipo de señal es la que se obtiene de la electrooculografía (EOG) que en sí es el potencial bioeléctrico que es generado al momento de realizar algún movimiento ocular [4, 5]. El EOG suele ser implementado en pruebas para la detección de trastornos oculares, pero otra utilidad que se le puede dar es estimar el posicionamiento de la vista debido a movimientos sacádicos [6], dado que la amplitud del potencial eléctrico generado con cada movimiento es directamente proporcional al desplazamiento realizado, a esta señal se le conoce como EOG DC.

Es posible estimar el posicionamiento de la vista de una persona en determinado momento utilizando el presente programa de cómputo desarrollado en MATLAB y conjuntos de señales de un patrón de movimientos previamente adquiridas y almacenadas. Es importante mencionar que las señales utilizadas para el desarrollo fueron creadas artificialmente representando el comportamiento de movimientos sacádicos generados por los ojos de un sujeto de prueba en condiciones ideales para validar el buen funcionamiento de la lógica de los algoritmos para el procesamiento, y posteriormente se evaluó y validó con señales reales.

El programa de cómputo carga dos conjuntos de datos, correspondientes a las mediciones de señales EOG de un sujeto de prueba, uno para calibrar la señal (obteniendo

los valores de voltaje máximos y mínimos extraídos con el sistema de adquisición y otros parámetros requeridos para la validación del código), y un segundo conjunto de datos con el que se utilizan los datos de calibración para estimar el posicionamiento ocular de un patrón de movimientos (para la comprobación del posicionamiento de la vista y visualización de forma gráfica). Aunque puede también trabajar con un solo conjunto de datos que incluya tanto lecturas de patrones para calibración, como posteriormente las lecturas de la medición. Cada conjunto de datos representan dos señales, estando constituido por dos columnas, en donde la primera de ellas contiene información de la señal en su forma de DC y la segunda columna sé encuentra en AC. Obteniendo como resultado una gráfica representando la estimación de la posición de la vista en un campo de visión a través del tiempo, con distancias establecidas por el número de píxeles en una pantalla.

1.1. Planteamiento del problema

Diseñar una interfaz para la interpretación de los movimientos oculares, adquiriendo señales de un sistema EOG, aplicable al control de dispositivos actuadores o sistemas que ayuden a personas con discapacidad motriz, proporcionándoles una comunicación manos libres. El interés en aplicaciones de detección de movimiento ocular ha incrementado recientemente [7]. Algunos métodos de detección ocular han sido utilizados en diversas aplicaciones tales como juegos de computadora, tecnologías de monitoreo para detectar la fatiga de un conductor de transporte público o comercial [8–10], en la industria de la publicidad, en la identificación de personas basada en el reconocimiento facial y la detección del iris ocular [11, 12], y en diferentes aplicaciones militares para ayudar a los pilotos a apuntar armas simplemente mirando un objetivo. Las aplicaciones de mayor impacto incluyen la neurociencia y psicología [13]. Las tecnologías de asistencia para comunicarse con pacientes discapacitados o tecnologías que les proveen de controles para manejar dispositivos digitales como computadoras o dispositivos electromecánicos tal como sillas de ruedas [14–17]. Estas últimas aplicaciones son de sumo interés, ya que una interface de interpretación del movimiento ocular puede ser utilizada

en aplicaciones para operar dispositivos que mejoren el entorno del paciente discapacitado y controlar actuadores a través de movimientos oculares; sin embargo, con el avance actual, la capacidad de control de actuadores a partir de movimientos oculares se limitaba principalmente a la detección de cuatro desplazamientos (izquierda, derecha, arriba y bajo), limitando las aplicaciones. Con el desarrollo de esta tesis se pretende obtener la detección de las coordenadas de un campo de visión que permitan manejar interfaces con al menos más de cuatro opciones de desplazamiento visual, fue parte de esta investigación el desarrollar una interfaz para la interpretación del movimiento ocular a partir del procesamiento de señales de EOG de movimientos horizontales, no solo se detecta el sentido del movimiento hacia izquierda o derecha sino que se detectan movimientos en intervalos en el eje horizontal, lo cual es un predecesor para el procesamiento de señales de EOG de movimientos verticales, y el cálculo de las coordenadas en movimientos oculares no solo en horizontal y vertical, sino en diagonal o en cualquier patrón de movimiento realizado a través de un campo de visión definido.

1.2. Justificación

El seguimiento ocular es el proceso de medir el punto de fijación de la mirada o el movimiento de un ojo en relación con la cabeza considerada fija regularmente. Un seguidor de ojos, es un dispositivo para medir la posición y el movimiento de los ojos [18–20]. Según la literatura, los principales métodos de seguimiento ocular se basan en, oculografía infrarroja (IROG, de sus siglas en inglés) [21, 22], videooculografía (VOG, de sus siglas en inglés) [23–26] y electrooculografía (EOG) [22,27,28]. Gracias al avance en el campo de la electrónica y los sistemas computacionales, las técnicas de detección de la dirección de la mirada se han desarrollado en dos direcciones básicas: procesamiento digital de imágenes mediante VOG [15] y procesamiento de señales mediante EOG [14]. En las técnicas de detección de la dirección de la mirada basadas en procesamiento digital de imágenes se presenta el inconveniente de procesamiento extenso y lento debido a que existen varios pre-procesos antes de llegar al análisis de la posición de la retina, tal como identificar el rostro y posteriormente identificar los ojos del rostro. Además,

exige un costo computacional elevado y el resultado de las interpretaciones en tiempo real es lento. Por otro lado, los movimientos oculares muestran respuestas primarias que reflejan la intención voluntaria y la selección consciente de los humanos. Debido a que la percepción visual es una de las interacciones sensoriales fundamentales en el cerebro, los movimientos oculares contienen información crítica sobre la salud física/psicológica, la percepción, la intención y la preferencia. Con el avance de las tecnologías de dispositivos portátiles, el rendimiento del monitoreo del seguimiento ocular ha mejorado significativamente. También ha dado lugar a innumerables aplicaciones para ayudar y aumentar o mejorar las actividades humanas. Entre ellos, los electrooculogramas, medidos con electrodos montados en la piel, se han utilizado ampliamente para tratar de seguir con mayor precisión los movimientos oculares. Por lo anterior, el resultado de este proyecto es el desarrollo de una interfaz para la interpretación de los movimientos oculares sacádicos horizontales a partir del procesamiento de señales EOG. La interfaz de caracterización e interpretación del movimiento ocular, pudiera ser utilizada para la investigación y desarrollo de aplicaciones que requieran operar y controlar actuadores a través de movimientos oculares. Es un método más económico comparado con métodos que utilizan cámaras o lentes especiales. El procesamiento de señales EOG tienen menor tiempo de respuesta comparado con métodos que usan procesamiento de imágenes. Es menos invasivo que otros métodos y no afecta la visión, ya que no usa luz infrarroja o cercana a infrarroja. Puede proporcionar una interface manos libres para personas con discapacidad motriz.

1.3. Objetivos

El propósito es evaluar los algoritmos para el procesamiento de señales EOG en el entorno de MATLAB. Con el objetivo de migrarlos a un entorno de desarrollo en LabView que permita instrumentar el sistema completo, desde la adquisición, el procesamiento y la representación de resultados, así como, la activación de actuadores de ser requerido por una aplicación. El objetivo de los algoritmos de procesamiento de señales EOG es estimar el posicionamiento de la vista con el uso de señales de EOG suprimien-

to el existente problema de la línea base, de ruido y la deriva de la señal en su forma de DC, ya que la amplitud de la señal idealmente es proporcional al desplazamiento del movimiento ocular. La estrategia de los algoritmos de procesamiento, consiste en utilizar las señales de EOG en su forma de AC a partir de los picos detectados en cada cambio de sentido del movimiento ocular para reconstruir una señal con un comportamiento similar al de DC pero disminuyendo problema existente de ruido y deriva de la señal original de EOG en DC, esto con la finalidad de estimar el posicionamiento de la vista en coordenadas en formato de píxeles del campo de visión (para movimientos horizontales, la dimensión del desplazamiento en el eje horizontal).

1.3.1. Objetivo general

El objetivo principal de este trabajo, es el desarrollo de interfaz para la interpretación de los movimientos oculares a partir del procesamiento de señales EOG para ser utilizado como heerramienta para el desarrollo de aplicaciones que requieran operar y controlar actuadores a través de movimientos oculares.

1.3.2. Objetivos específicos

- **Primer Objetivo:** Conocer los fundamentos y el estado del arte en la adquisición, procesamiento, y la interpretación de señales EOG que correlacionan los movimientos oculares, para definir las mejores estrategias utilizadas, los alcances que se han obtenido y las limitaciones que requieren ser resueltas, así como describir su utilidad y las áreas de aplicación.

- **Segundo Objetivo:** Migrar la circuitería electrónica del sistema de adquisición de señales EOG en una PCB para robustecer el sistema.

- **Tercer Objetivo:** Implementar la adquisición y almacenamiento de señales EOG para post-procesamiento de pruebas preliminares de algoritmos.

- **Cuarto Objetivo:** Definir patrones de interés de movimientos oculares sacádicos.

- **Séptimo Objetivo:** Diseñar y programar una interfaz para la interpretación de los movimientos oculares a partir de la adquisición, calibración y procesamiento de señales EOG, a fin de proveer de una herramienta para su aplicación en la resolución de problemas reales.
- **Octavo Objetivo:** Realizar la publicación de resultados científicos en la presente tesis.

1.4. Hipótesis

Es posible correlacionar las señales EOG con el movimiento ocular, a través del procesamiento de señales para el diseño y programación de una interfaz que podrá ser implementada en diversas aplicaciones para conocer los posicionamientos de la mirada con el objetivo de controlar actuadores.

1.5. Antecedentes

El interés en el desarrollo de interfaces de interacción humano-computadora (HCI) ha incrementado en las últimas décadas [29] para trabajar con dispositivos digitales e informáticos a modo de manos libres en campos como realidad aumentada [30, 31], juegos [32–34], cirugía [35] y ayuda a personas con discapacidad como el caso de la esclerosis lateral amiotrófica avanzada (ELA), la cual es una enfermedad progresiva del sistema nervioso que afecta las neuronas en el cerebro y la médula espinal, y causa pérdida del control muscular [36].

Existen dispositivos que permiten la adquisición de señales fisiológicas. La mayoría de ellas con un nivel de maduración científica alta en la interpretación de su morfología y correlación con padecimientos de la salud, pero son escazas la caracterización e interpretación de las señales EOG que las relacione con el movimiento ocular y el posicionamiento de la fijación de la vista y persecución de objetos [37]. Estos dispositivos tienen una respuesta en frecuencia que excluye los componentes de bajas frecuencias (considerados como artefactos) porque dificultan su interpretación; pero en el caso de

la señal EOG es importante, ya que su ancho de banda se encuentra entre los 0– 50 Hz, varios de los sistemas de adquisición propuestos en la literatura atenúan los artefactos de bajas frecuencias afectando la morfología de la señal y limitando su resolución de identificación de movimientos oculares. En esta investigación se utilizó un sistema de adquisición de señales EOG que conserva su contenido de frecuencia de DC-30 Hz para incrementar la resolución de las interfaces para las tecnologías de interacción humano-computadora; no solo detectar el sentido del movimiento ocular, sino también puede contabilizar su desplazamiento e identificar la posición aproximada de su fijación, para mapear un campo de visión con las coordenadas (en el eje horizontal) donde se fijó la vista, generando herramientas para el desarrollo de interfaces de interacción humano-computadora para proveer a las personas con discapacidad motriz la manipulación de ambientes computacionales para trabajar con dispositivos digitales e informáticos a modo de manos libres como realidad aumentada, juegos, cirugía y ayuda a personas con discapacidad como lo es el caso de la esclerosis lateral amiotrófica avanzada (ELA), la cual es una enfermedad progresiva del sistema nervioso que afecta las neuronas en el cerebro y la médula espinal, y causa pérdida del control muscular.

Marco teórico

2.1. Investigación del estado del arte

Actualmente, el valor del análisis y evaluación del movimiento ocular es reconocido para diversas aplicaciones [37]. Ya que revela información de la que se pueden extraer patrones acerca del enfoque de las personas en diversos contextos [3], como la fatiga cognitiva [38, 39], de salud mental, salud ocular como el estrabismo, problemas con la retina y su capacidad de adaptarse a cambio en la iluminación ambiental [40], trastornos de sueño [41, 42], de atención, intereses, y patrones de navegación a través de un marco de referencia de espacio-tiempo, también para autentificación inclusiva de personas que no pueden autentificarse con huellas, rostro o el uso de la memoria para códigos, debido a discapacidades de movilidad y a deterioro cognitivo [43], pueden ser utilizados también como detectores de somnolencia para conductores de vehículos [44]. Además, de que pueden ser utilizados como controladores de actuadores a través de una interfaz de interacción humano-computadora, en especial para personas con movilidad limitada [45]. Así como, interfaces para sistemas de realidad virtual [46].

Entre los enfoques de procesamiento de señales EOG para la eliminación de ruido debido a señales del entorno como frecuencias del voltaje de línea (50/60 Hz), artefactos de cables debido a movimientos y artefactos musculares (EMG), se encuentran los basados en métodos de filtrado, y los basados en la eliminación de ruido orientada a modelos [5, 47]. Los enfoques basados en aprendizaje máquina y aprendizaje profundo se centran en determinar los patrones de la señal para interpretar el contenido de dichas

señales [6]. Aunque en los sistemas actuales de adquisición de EOG enfrentan desafíos para garantizar la comodidad del usuario, particularmente en términos del rendimiento eléctrico y mecánico de los electrodos, la usabilidad a largo plazo, los efectos térmicos y la portabilidad general del sistema.

2.2. El ojo humano y la electrooculografía

El epitelio pigmentario de la retina (EPR) es una capa de células, situada por fuera de los fotorreceptores, que contribuye a mantener la retina externa. Entre sus funciones está absorber la luz sobrante para evitar que los fotorreceptores se reestimulen, transportarles el retinol en el ciclo de los fotopigmentos, fagocitar los discos que se desprenden del segmento externo de los fotorreceptores, transportar fluidos y metabolitos, regular el contenido del citosol y, también, forma parte de la barrera hemato-retiniana externa. El EPR es responsable de la diferencia de potencial existente entre la córnea y el fondo del ojo, formando un dipolo en el eje óptico, positivo en la córnea y negativo en el fondo. Este potencial se puede registrar mediante electrodos colocados en la piel cerca del ojo ya que varía según el ángulo de rotación del ojo, hecho que se utiliza para estudiar la motilidad ocular. Además, el potencial córneo-fundal del ojo a oscuras se modifica considerablemente con la luz. El registro del potencial y su comportamiento en oscuridad y con luz es la base del electrooculograma [48–50].

2.3. Sistema de adquisición EOG

El sistema de adquisición de señales EOG mide el potencial eléctrico generado por el movimiento ocular, ya que la córnea y la retina forman un dipolo eléctrico. Este sistema incluye etapas de amplificación y filtrado para eliminar ruido antes de digitalizar la señal. Se usa un amplificador de instrumentación por su precisión y capacidad de rechazar interferencias. Para eliminar el ruido de la red eléctrica, se emplea dos filtro Notch, y para reducir otras frecuencias no deseadas debido a artefactos, se utilizan filtros activos pasa-bajas y pasa-altas. Estos componentes permiten obtener señales limpias y

útiles para procesarse digitalmente a fin de interpretarse.

2.3.1. Sistema de adquisición de señales EOG

El sistema de adquisición EOG es un dispositivo que es capaz de medir el potencial eléctrico generado por el movimiento ocular dado que la cornea y la retina funcionan como un dipolo eléctrico; al momento de mover los ojos a alguna dirección, la diferencia de potencial varía [51].

El diseño de un sistema EOG conlleva distintas etapas, una de ellas es la amplificación de la señal entrante y la otra es el filtrado (ver Figura 2.1) para la atenuación de señales indeseadas, para posteriormente ser adquiridas por una tarjeta de adquisición de datos ya sea para ser almacenadas o procesadas durante la adquisición.

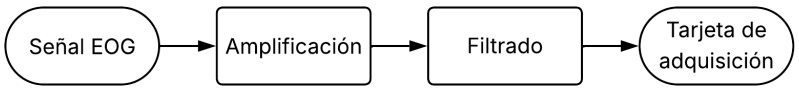


Figura 2.1: Proceso para la lectura de señales de EOG.

En la Figura 2.1 se aprecia el flujo que comúnmente es utilizado para el diseño de un sistema de EOG. La información entrante al sistema de adquisición es proveniente de un sujeto generando movimientos oculares, posteriormente la información es amplificada y filtrada, para finalmente adquirir y convertir la señal a digital y procesarla en un ordenador [51, 52].

2.3.2. Amplificación de un biopotencial

Los biopotenciales que son generados por el movimiento ocular se encuentran en un rango que va de los 10 a los 3500 μV y se encuentran en un rango de frecuencias que parten de DC a los 100 Hz [53,54], otra fuente menciona que las señales que representan los movimientos sacádicos parten desde DC hasta los 30 Hz con una amplitud de 100 a 2000 μV [55], los movimientos registrados en este rango son los movimientos de persecución lentos y sacádicos (descatando los microsacádicos), parpadeos (con poca definición).

Debido a que los biopotenciales son débiles en comparación con otras señales, se suelen utilizar los amplificadores de instrumentación para amplificar dichos potenciales diferenciales generados por el movimiento ocular. El amplificador de instrumentación es un circuito integrado creado a partir de amplificadores operacionales en modo diferencial, por lo que cuenta con una alta precisión y estabilidad, además de que el ajuste del valor de la ganancia es más directo y sencillo. Los amplificadores de instrumentación suelen ser menos susceptibles al ruido debido a su alto valor en su Factor de Rechazo en Modo Común (CMRR) y ademas tienen una alta impedancia de entrada, captando así gran parte de la señal. En la Figura 2.2 se muestra el símbolo electrónico del amplificador de instrumentación INA128, donde V_{IN^-} y V_{IN^+} son la diferencia de potencial de entrada que será amplificada, R_G la resistencia que se utiliza para ajustar la ganancia de amplificación, Ref es el común y V_O es la salida amplificada [56].

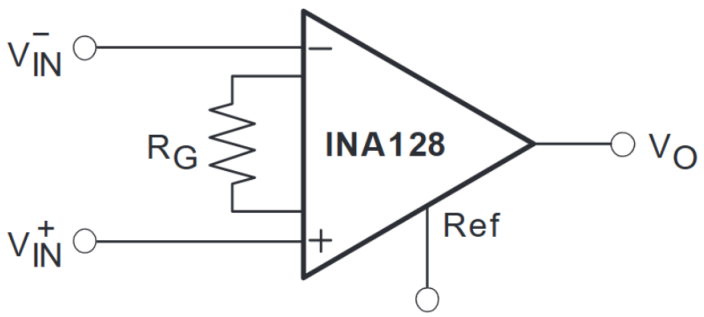


Figura 2.2: Diagrama del amplificador de instrumentación INA128 [62].

El componente R_G ajusta la ganancia (G) de salida en V_O. Esta se puede calcular con la Ecuación 2.1.

$$G = 1 + \frac{50k\Omega}{R_G} \tag{2.1}$$

Las terminales V_{IN^-} y V_{IN^+} registran el biopotencial generado por el movimiento ocular, amplificando G veces la diferencia de potencial entre las terminales. El cuerpo humano también puede actuar como una antena que capta interferencias electromagnéticas, especialmente el ruido de 50/60 Hz de las líneas eléctricas. Esta interferencia puede oscurecer las señales biológicas, dificultando su medición. El circuito de control de pierna derecha se utiliza para eliminar el ruido de interferencia mediante su

cancelación activa [57]. El circuito de pierna derecha es mostrado en la Figura 2.3, en el pueden observarse las terminales V_{IN^-} y V_{IN^+} conectadas como se mencionó anteriormente, además de eso, la tercer terminal (Ref) esta conectada directamente a la resistencia de 390 k_Ω que está conectada a la salida del primer OPA2131.

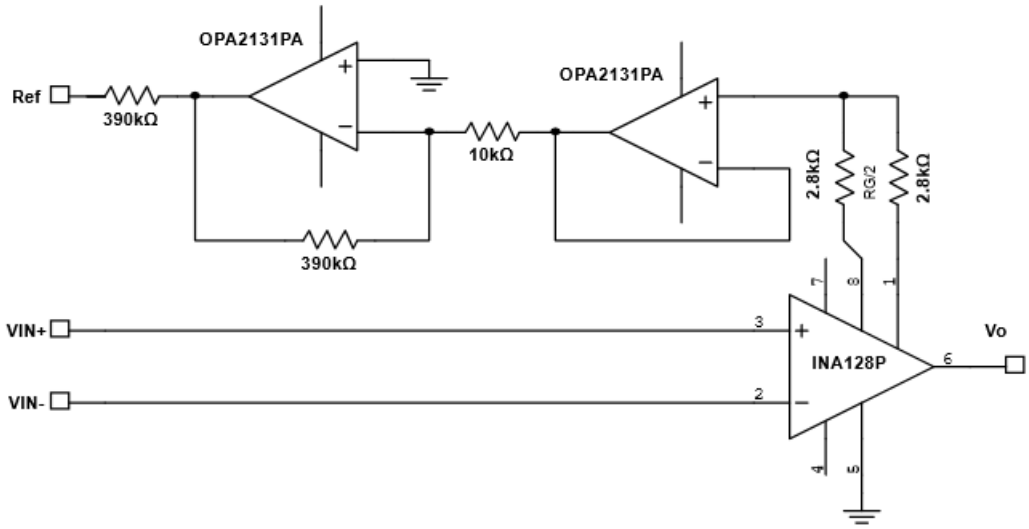


Figura 2.3: Circuito de pierna derecha conectado al amplificador de instrumentación.

2.3.3. Filtro Notch

Una de las variantes más utilizadas de los filtros rechaza banda es el filtro Notch, que tiene una banda de rechazo estrecha, muy útil para eliminar componentes específicos como frecuencias no deseadas de una señal, sin comprometer las frecuencias restantes de interés. Su uso es muy común para eliminar el ruido generado por la red eléctrica, que funciona a 50 o 60 Hz [58].

El UAF42 es un filtro activo continuo en el tiempo y de segundo orden para diseños de filtros complejos y simples. Utiliza la arquitectura analógica clásica de variable de estado con un amplificador sumador y dos integradores. El UAF42 puede usarse como filtro Notch de 60 Hz para la atenuación efectiva de esa frecuencias con el uso externo de 6 resistencias. En la Figura 2.4 se presenta la configuración del UAF42 para suprimir la frecuencia de línea (60 Hz) [59].

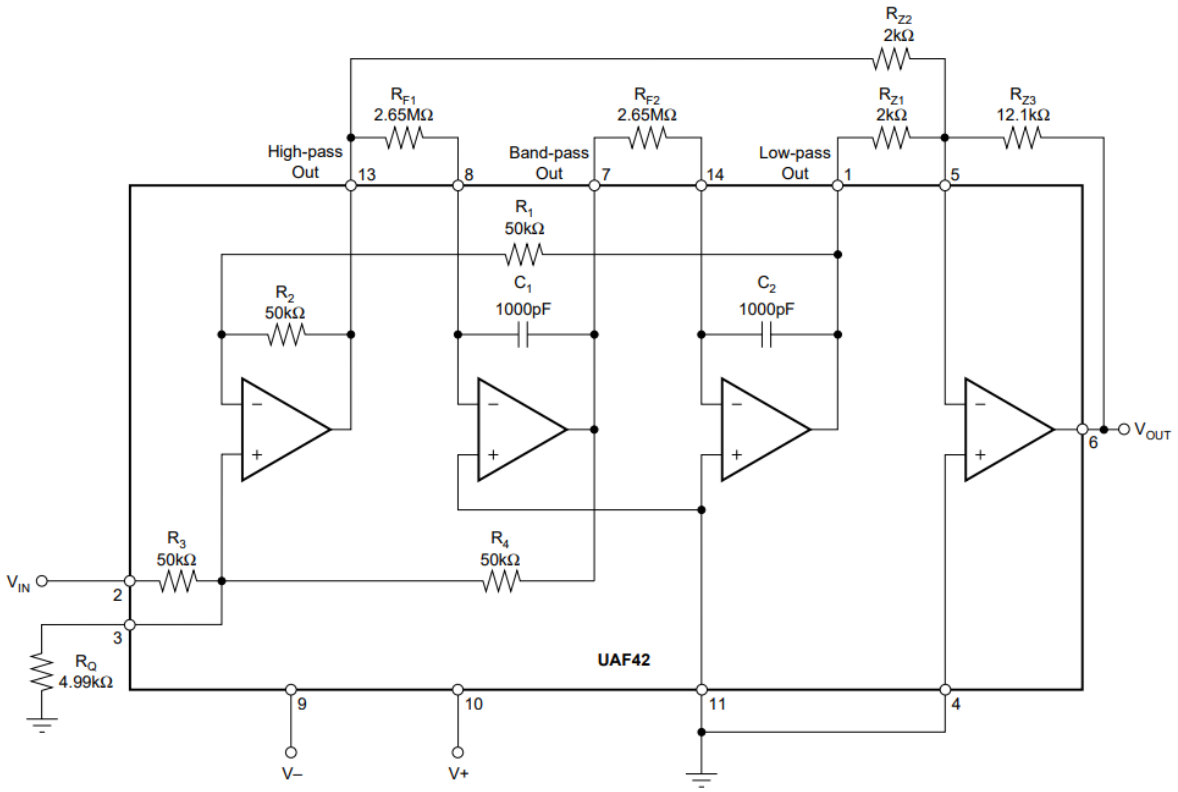


Figura 2.4: UAF42 configurado como un filtro Notch de 60 Hz [65].

2.3.4. Filtro activo pasa-bajas

Los filtros activos pasa-bajas son útiles para la atenuación del ruido de alta frecuencia. Un filtro activo pasa-bajas con arreglo Sallen Key se consigue utilizando el amplificador operacional como se muestra en la Figura 2.5.

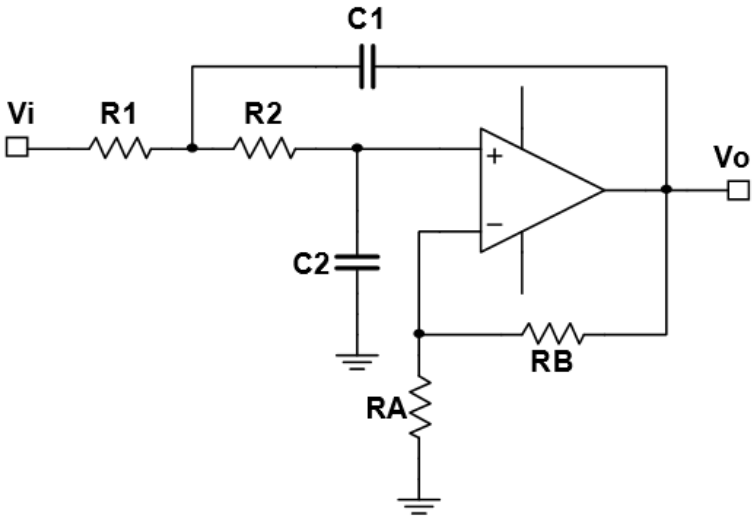


Figura 2.5: Filtro activo pasa-bajas Sallen Key.

Las ventajas de este circuito es que es capaz de tener ganancia y una salida de muy baja impedancia. La respuesta de frecuencia viene dada por la relación entre la retroalimentación y la impedancia de entrada.

La ganancia en la salida se calcula con la Ecuación 2.2

$$k = RA + \frac{RB}{RA} = 1 + \frac{RB}{RA} \tag{2.2}$$

Para determinar la frecuencia de corte (f_c), es posible elegir valores de $C1$ y $C2$ iguales y de igual manera, hacer lo mismo para las resistencias $R1$ y $R2$, de este modo, la frecuencia de corte puede ser calculada utilizando la Ecuación 2.3.

$$f_c = \frac{1}{2\pi RC} \tag{2.3}$$

2.3.5. Filtro activo pasa-altas

En la Figura 2.6 se representa un amplificador como filtro activo pasa-altas con configuración Sallen Key, que es usado comúnmente para ampliar y no permitir el paso de señales en frecuencias menores para el que fue diseñado. La ventaja de este filtro, al igual que el pasa-bajas Sallen Key, es que en la frecuencia de corte, la señal es atenuada en una proporción de $\frac{1}{\sqrt{2}}$ y cuanto más se aleje del valor de la frecuencia de corte (lejos

del rango de las frecuencias permitidas), la señal se atenúa hasta ser insignificante.

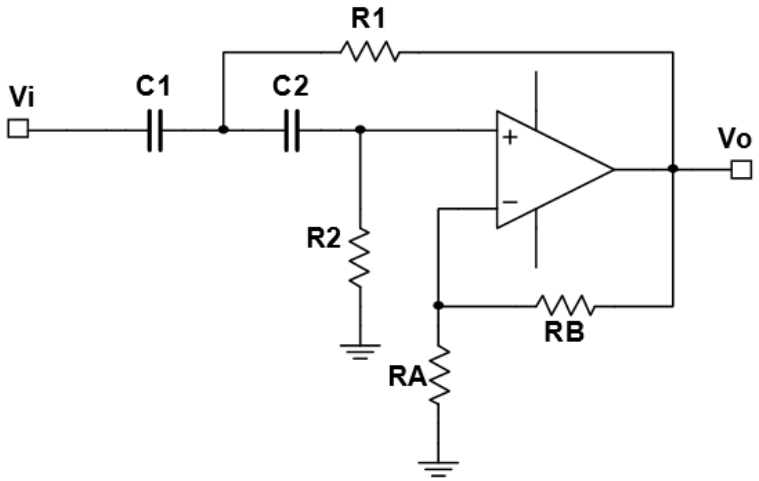


Figura 2.6: Filtro activo pasa-altas Sallen Key.

Los cálculos para la ganancia y para las frecuencias de corte, son los mismos que se utilizan con el filtro pasa-bajas visto en la Sección 2.3.4. La principal diferencia entre los filtros pasa-altas y pasa-bajas es el tipo de respuesta ante las distintas frecuencias debido a la disposición de las resistencias $R1$, $R2$ y los capacitores $C1$, $C2$.

Metodología

En esta sección se presenta la metodología que describe el procedimiento realizado en el diseño de una interfaz para la interpretación del movimiento ocular a partir del procesamiento de señales de EOG.

Se interpreta el movimiento sacádico ocular en horizontal utilizando señales de EOG en su forma de AC (resultado del filtro pasa alta), estimando y reconstruyendo la señal de origen en DC (resultado del filtro pasa baja) para atenuar los efectos de la deriva persistente de la señal original en DC.

Para esto se utilizó el entorno de programación Matlab, a través de una aplicación con interfaz de usuario que permite adquirir señales en un aproximado al tiempo real o trabajar con señales previamente adquiridas y almacenadas.

La aplicación permite configurar los parámetros de operación, para dar paso al procesamiento de la señal, y finalmente desplegar la interfaz para la interpretación del movimiento ocular a partir del procesamiento de señales de EOG adquiridas.

La aplicación desarrollada estima el posicionamiento ocular después de realizar movimientos sacádicos horizontales en un campo de visión predefinido, lo cual requiere una caracterización y configuración tanto de las dimensiones de los límites del campo de visión, así como de la interfaz donde se desplegarán los resultados de interpretación de las señales adquiridas.

Para el desarrollo del código de procesamiento de las señales en Matlab, fue útil simular la adquisición dato a dato de la señal a procesar a partir de datos sintéticos. Primero, con el objetivo de acondicionar el código para ser utilizado en la adquisición y procesamiento en tiempo real y segundo para validar los resultados de las operaciones

dentro de las etapas de calibración y procesamiento de la señal.

Una vez validado el funcionamiento del procesamiento de las señales EOG a partir de la señal de AC, para representar el movimiento ocular en la forma de señal DC, ya que la amplitud de la señal de DC sin deriva es directamente proporcional al desplazamiento ocular; dicha representación es reflejada en la interfaz de la aplicación desarrollada en un valor de voltaje y/o en coordenadas en píxeles, las cuales pueden ser relacionadas con dimensiones de desplazamiento en centimetros de acuerdo a la caracterización del campo de visión.

Una vez que la aplicación fue probada con señales sintéticas simulando el comportamiento de movimientos oculares en un escenario ideal, también fue puesta a prueba con señales reales adquiridas con el prototipo de adquisición de señales de EOG a personas realizando patrones de movimientos oculares sacadicos horizontales, almacenadas para su evaluación.

3.1. Descripción General del Procesamiento

En esta sección se escribe de forma general, para una visión global, el flujo de procesamiento asumiendo que ha iniciado la adquisición de la señal de AC. En las siguientes secciones se desarrollan los detalles técnicos de cada subproceso.

A continuación se describe el diagrama de flujo para procesamiento de datos de señales EOG en su forma AC para estimar la señal en DC a partir del procesamiento dato a dato representado en la Figura 3.1.

a) "INICIO", inicia el programa para posteriormente ejecutar funciones.

b) "Ajuste de parámetros", se establecen parámetros de inicio para la detección de picos (los picos representan cambios en el movimiento ocular sacádico horizontal) como lo son la altura del umbral positiva y negativa +/- (la amplitud representa la dimensión del desplazamiento, y el signo la dirección derecha-izquierda o viceversa, depende de la colocación de los electrodos), cantidad de datos necesarios para realizar una nueva detección (para que el umbral pueda volver a detectar un nuevo pico) y la cantidad de datos para la ventana del suavizado de media móvil con el objetivo de atenuar el

ruido en la señal en procesamiento. En el caso de la calibración, este proceso es el b) presentado en la figura 3.1.

c) “Procesamiento (Realizar calibración)”, una vez con el ajuste completo de parámetros de inicio, se procede a la obtención los parámetros de calibración, que serán utilizados para la reconstrucción de la señal DC que representara los movimientos oculares, en la Figura 3.4 es descrito a detalle el procesamiento para obtener los parametros de calibración.

d) “Obtención de parámetros de calibración realizada”, una vez completado el procedimiento de calibración, quedan definidos los parametros de calibración para su futura utilización.

e) “Utilizar los parámetros establecidos en la etapa de calibración”, se cargan los parámetros de calibración que corresponden a la configuración de adquisición de señales para las que se iniciara el proceso de interpretación de movimientos oculares a partir de la señal en AC.

f) “Procesamiento (Realizar aproximación)”, se realiza el procesamiento de la señal AC para obtener una reconstrucción aproximada de la señal DC (sin deriva), dicho bloque y su funcionamiento es detallado en la Figura 3.5, con el cual se obtiene la aproximación (reconstrucción DC, interpretación de movimientos oculares).

g) “Resultados en volts y coordenadas en píxel”, el procesamiento se realiza dato a dato y al finalizar el proceso que detecta un cambio de posición ocular se pueden ver los resultados del movimiento anterior en la ventana de datos completa, los resultados son desplegados en volts y coordenadas en píxeles de un campo de visión definido.

h) “Guardar señal”, el sistema ofrece la opción de guardar la señal resultante para posterior análisis de resultados.

i) “Guardar el resultado en voltaje y en píxeles en un archivo .csv ”, en el caso de querer guardar el arreglo resultante, se creará un archivo en el formato “.csv” con el resultado obtenido y con el nombre asignado a dicho archivo.

j) “FIN”, finaliza al detener el programa manualmente durante la ejecución del mismo.

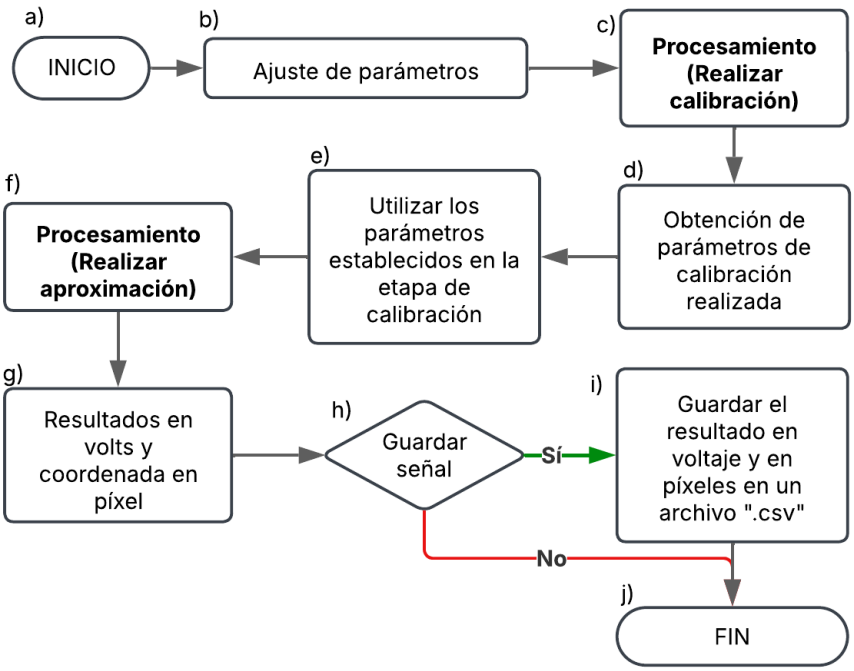


Figura 3.1: Diagrama de flujo para procesamiento de datos de señales EOG en su forma AC para estimar la señal en DC a partir del procesamiento dato a dato.

Este procesamiento se lleva a cabo a través de la aplicación desarrollada, la cual es una interfaz para visualizar la interpretación del movimiento ocular a partir del procesamiento de señales de EOG, y cuenta con elementos de interfaz de usuario, para darle los comandos necesarios para su operación, ver Figura 3.2.

Las instrucciones para la instalación del entorno de programación para el desarrollador se describen en Sección 4.3. Las instrucciones para la instalación de la aplicación del programa de cómputo para el usuario se describen en la Sección 4.1.

El programa cumple su propósito con dos modos de uso, el primero de estos modos es descrito en Sección 3.4 (se utiliza en solo valor de umbral para la detección de picos a fin de calibrar y aproximar la señal), en el cual es estrictamente necesario contar con dos archivos ".csv", uno de ellos para calibración y otro para evaluar el movimiento ocular, siendo en el primer modo donde se utiliza un solo umbral para obtener la calibración, se usará para obtener el promedio de crestas y valles (picos máximos y mínimos) en la señal AC para emplearlos en la representación en voltaje y convirtiéndolos a una coordenada

en píxeles representativa al campo de visión y la evaluación de la reconstrucción de la señal. El segundo modo es descrito en la Sección 3.5 (utilizar un archivo de evaluación para calibrar segmentando los picos que correspondan a movimientos de centro a extremo o viceversa, además existe un umbral nuevo para ajustarlo nuevamente para ignorar los picos mas pequeños que pudieran ser validos pero a la vez ser confundidos con ruido o parpadeos), donde puede haber un archivo de calibración independiente o puede no haberlo, aquí se utilizan dos umbrales distintos, el primero de los umbrales se usará para obtener el promedio de crestas y valles (picos máximos y mínimos) en la señal AC para emplearlos en la representación en voltaje y convirtiéndolos a una coordenada en píxeles representativa al campo de visión; y el segundo umbral es útil para establecer los desplazamientos sacádicos (amplitud de voltaje, picos) que serán representados en la reconstrucción de la señal. Para el ejemplo del modo dos, se utilizará solamente una señal tanto para la calibración como la evaluación, separando los picos máximos para obtener los promedios representativos de los márgenes y la evaluación, contemplando todos los picos de la señal.

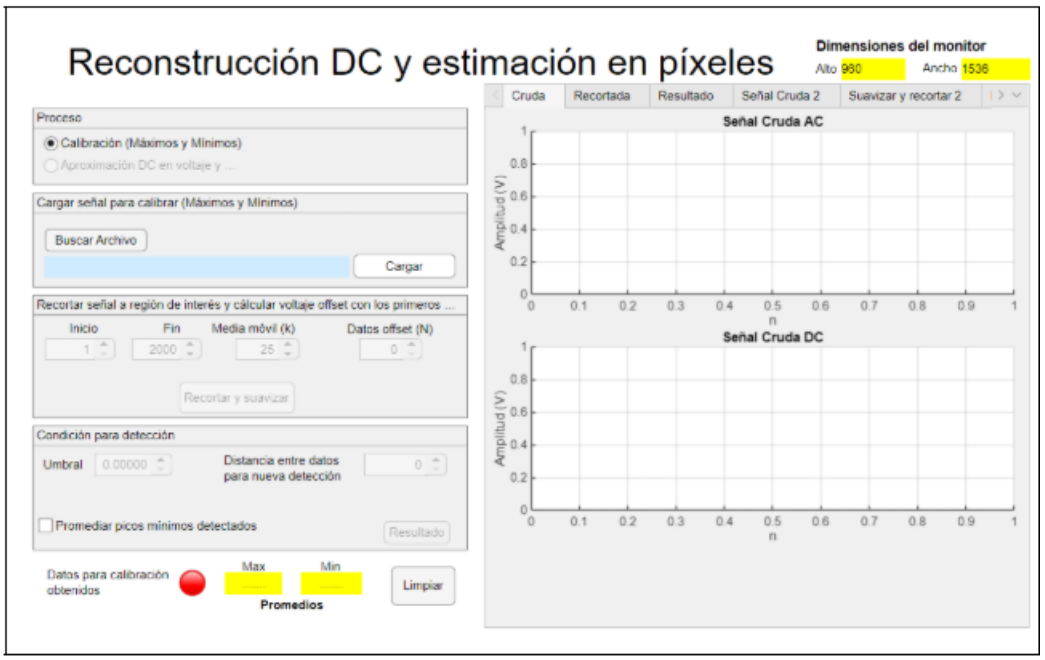


Figura 3.2: Interfaz para la interpretación del movimiento ocular a partir del procesamiento de señales de EOG.

3.2. Proceso de Calibración

El objetivo de esta etapa es obtener dos valores que representan el desplazamiento ocular del centro a un extremo del campo de visión y del centro hacia el otro extremo. Esto se logra siguiendo una rutina de movimientos sacádicos de un punto en una pantalla; ese seguimiento es realizado las veces que sean necesarias para obtener un buen promedio en los valores y, en base a eso, obtener una mejor estimación de los valores representativos de las amplitudes del centro a un extremo de la pantalla. Para entender mejor lo mencionado, véase la Figura 3.3, en donde el centro es el punto B, y los extremos son los puntos A y C; estos valores se utilizarán en una futura etapa de evaluación o estimación del posicionamiento de la vista.

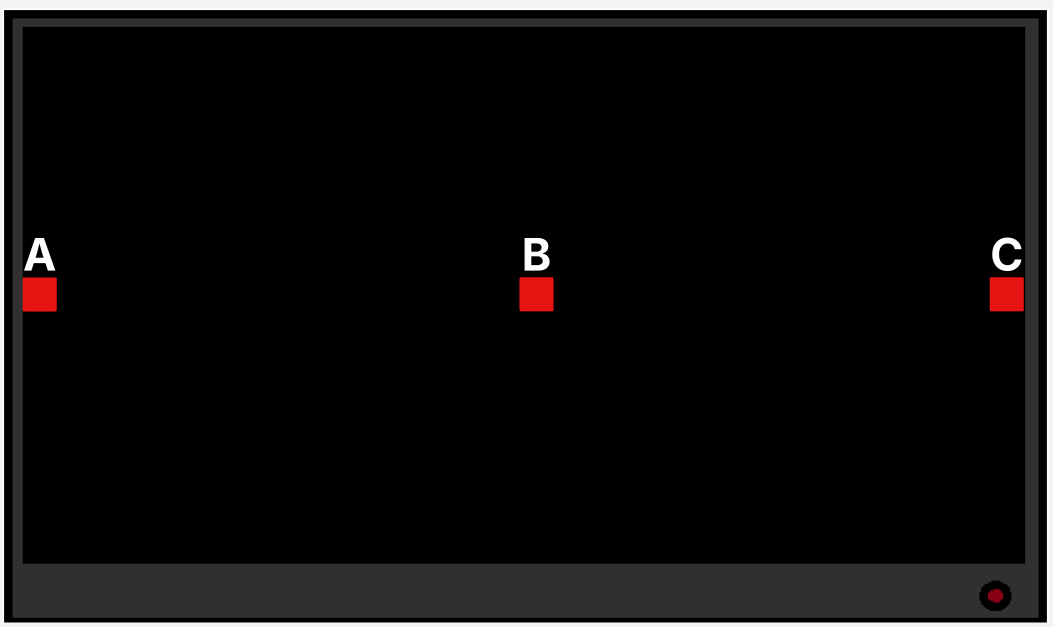


Figura 3.3: Representación del campo de visión y puntos a observar para calibrar y obtener los valores máximos y mínimos de distancia en píxeles de la pantalla en base al posicionamiento ocular.

En la Figura 3.4, se muestra la etapa del procesamiento del algoritmo encargado de la obtención de los parámetros máximo y mínimo de la pantalla respecto al movimiento ocular. Este procesamiento es la etapa c) de la Figura 3.1. A continuación, se describe el proceso de la Figura 3.4:

a)"INICIO", se inicia con la definición de parámetros manualmente, en donde se configura la variable c (representa la cantidad de datos que deben transcurrir para

realizar una nueva detección de movimiento ocular), n (cantidad de datos necesarios para crear el arreglo para el cálculo del voltaje offset), k (el tamaño de la ventana para el suavizado por media móvil) y u (el valor de umbral establecido para la detección de los picos).

b) "Lectura de dato", al tener los parámetros establecidos, se procede a leer dato a dato la información proveniente de la señal.

c) "¿Cantidad de datos leídos superan el valor de la ventana k?", aquí se verifica la condición sobre si el valor en posición del dato x_i es mayor o igual a k. En el caso de no cumplirse dicha condición, se regresará a la etapa b) y se leerá un nuevo dato (los datos anteriores permanecen en un arreglo de tamaño k); por otra parte, de cumplirse la condición, se procede a ejecutar el proceso d).

d) "Suavizado por media móvil con ventana de tamaño k", al cumplirse la condición c), se realiza un suavizado de media móvil utilizando el dato actual entrante x_i y los datos $k-1$ anteriores. La salida resultante de la operación es un dato suavizado con el ruido de alta frecuencia atenuado, el valor de este es almacenado en la variable xs_i.

e) "¿Datos suficientes para calcular offset?", se verifica si se ha alcanzado la cantidad de datos para realizar el cálculo del voltaje de offset; de no cumplirse la condición, se procede a almacenarse los datos en el arreglo descrito en f), por el contrario, si se cumple la condición, el flujo continúa en g).

f) "Crear arreglo de datos para offset", en esta etapa se crea un arreglo de nombre A de una dimensión y de tamaño n con los datos suavizados entrantes.

g) "¿Voltaje Offset ha sido calculado?", teniendo creado el arreglo A, se procede a verificar si ha sido calculado previamente el valor de voltaje offset de la señal EOG con los datos suavizados. Si no se cumple la condición, se ejecuta el proceso h); en caso de cumplirse, se procede a i).

h) "Cálculo del valor offset (voff)", en este proceso se realiza el cálculo del voltaje de offset promediando los datos pertenecientes al arreglo A, el resultado es almacenado en la variable xm.

i) "Restar voff al dato actual", al dato actual suavizado (xs_i) se le resta el dato calculado en h); obteniendo un valor sin el voltaje offset que contenía la señal anterior-

mente, de modo que los datos posteriores a esta etapa se encuentren montados en un valor aproximado a 0 volts. El resultado es almacenado en la variable xu.

j) "¿Estado actual del interruptor s?", posterior a obtener xu en el proceso i), es realizada la verificación del estado del interruptor digital s (es de estado booleano), en donde si este se encuentra en falso (F) se procederá a ejecutar la etapa k), en el caso de que se encuentre en verdadero (V) se realizará la etapa l).

k) "¿Supera el umbral?", se comprueba si el estado del valor actual suavizado xu_i, en donde, si este es menor al valor de umbral u, se omitirá el proceso l), regresando al estado inicial (si no se termina el programa). Si se cumple la condición de superar el umbral, se procederá a realizar l).

l) "Incrementar una unidad a d y encender interruptor s", al haber cumplido la condición de j) o k), se realiza el aumento en una unidad a la variable d (funciona como un contador) y se cambia el estado de s a V. Es importante mencionar que la variable d se encuentra con un valor de 0 inicialmente.

m) "¿Dato actual mayor a anterior?", en este punto se realiza una comparación entre el valor de xu_i y xy (la variable comienza con un valor inicial de 0, ya que es lo que se aproxima al valor de xm), de cumplirse la condición, el flujo se dirige al proceso n), de lo contrario el flujo irá al proceso o).

n) "Mantener dato actual", de cumplirse la condición descrita en m), será reemplazado el valor de xy por el de xu_i. Lo mismo ocurrirá siempre que se cumpla la condición, en futuras iteraciones el valor de xy irá cambiando al cumplir la condición.

o) "Mantener dato anterior", de no cumplirse la condición de m), el valor de xm se mantendrá como en el valor de xm anterior, descartando así lo que valía xu_i en ese momento, dado que no cumplió la condición de m).

p) "¿Número de datos para nueva detección superado?, se compara el valor del contador d contra el valor del parámetro definido c (cantidad de datos para realizar una nueva detección de umbral). De no cumplirse la condición, se realizará q), de otro modo, se hará r) al cumplirse la condición.

q) "Reiniciar xa a 0", el valor de la variable que almacena el pico más alto (xa) permanece intacto, posterior a este proceso se lee un nuevo dato proveniente de la

señal.

r) “Almacenar en un arreglo el pico máximo”, el dato contenido en la variable xy, se almacena en la variable xa en el instante i, en el proceso se almacena de tal forma que el valor de xy se escriba una sola vez cada que ocurra r), en futuras iteraciones al no ocurrir ocurre r), se asignará el valor de 0 a xa_i.

s) “Reiniciar d a 0 y apagar el interruptor s”, en esta parte, se ejecuta el reinicio de la variable contadora d a 0 y el interruptor s regresa al estado F.

t) “¿Finalizar?”, el operador de la aplicación es libre de detener el programa en cualquier momento, al decidir hacerlo, se realiza el proceso u), de lo contrario continua el proceso.

u) “Obtención de valores máximos y mínimos”, en esta etapa se calculan los valores promedio de los picos máximos positivos y negativos para su posterior uso en la etapa de evaluación.

v) “FIN”, finaliza el proceso de calibración.

Figura 3.4: Diagrama de flujo del funcionamiento del algoritmo elaborado para la obtención de valores máximos y mínimos del campo de visión en el proceso de calibración.

3.3. Proceso de Reconstrucción de Señal Aproximada a DC

A continuación se procede a describir una serie de pasos en donde el objetivo es estimar las coordenadas del posicionamiento ocular en voltaje y su equivalente en píxeles.

a) "INICIO", se inicia el programa utilizando los parámetros previamente obtenidos

en la etapa de calibración, dichos parámetros son útiles para realizar la aproximación a la señal de DC del EOG. Además de los parámetros con los cuales se inició la etapa de calibración, aquí también se suman los parámetros de máximo y mínimo de la pantalla que fueron obtenidos en la misma etapa.

b) "Lectura de dato", se procede a leer el dato "i" de la señal proveniente (la lectura es dato a dato).

c), ..., q) Lo actual corresponde a los procesos desde el "c" al "q", en donde se siguen una serie de pasos similares a los de la sección de calibración, esto se describe la Figura 3.4.

r) "Realizar aproximación de DC", en este proceso se realiza la estimación ajustando el valor del pico máximo a la variable xa, al contrario de lo que se hizo en el proceso de calibración, en la aproximación el valor de xa cambia cada vez que se supera el umbral sumando o restando el valor de xy y también es considerado el valor anterior de xa. Cuando es la primera vez que se le asignará un valor a xa, solamente será tomado en cuenta el valor de xy y no de xa anterior.

s) "Reiniciar d a 0 y apagar interruptor s", en esta etapa se restaura el valor de la variable contadora d y el interruptor s se restablece al valor de F. "Finalizar", en esta etapa se verifica si existe algún paro en el programa por parte del operador; en caso de ser afirmativa la verificación, se cambia la ruta al proceso u). "FIN", finaliza el proceso de aproximación. Caso contrario, continua en el proceso de adqusición en b) "Lectura de dato".

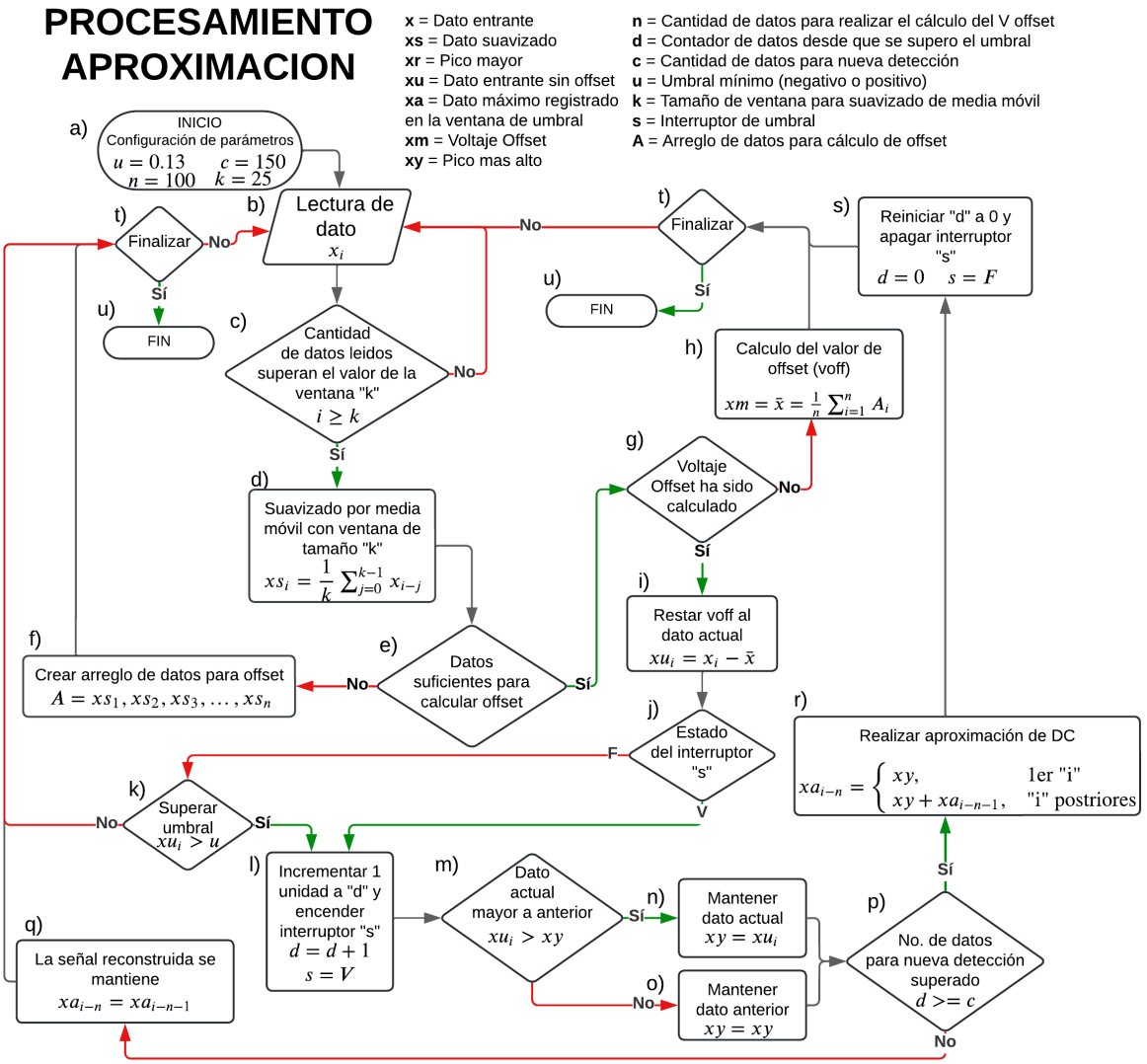


Figura 3.5: Proceso para realizar la aproximación de la señal en DC utilizando la señal AC del EOG.

3.4. Primer Modo de Funcionamiento

Para el primer modo de funcionamiento se emplea un solo valor del umbral para la calibración y la aproximación de la señal de AC a DC. Con el objetivo de obtener el promedio de picos máximos y mínimos en la señal AC, los cuales se utilizan para representar la señal en voltaje, convirtiéndolos a una coordenada en píxeles acorde al campo de visión y al proceso de reconstrucción de la señal DC.

El algoritmo fue probado con señales sintéticas representativas de movimientos EOG,

en donde la primera de ellas corresponde a observar del centro de un campo de visión hacia un extremo de ese campo, para posteriormente hacer lo mismo desde el centro hacia el extremo opuesto al anterior, delimitando así la posición de visión máxima permitida del campo de visión; las señales obtenidas de esta forma fueron las utilizadas para realizar la calibración del sistema (obtener valores de voltaje máximos y mínimos para convertir a coordenadas en píxeles en la estimación a DC).

Para estimar la señal DC a partir de una señal AC con la calibración anterior, se creó una señal sintética que representa un movimiento ocular sacádico, colocándose en distintos puntos de una pantalla, partiendo desde el centro hacia una distancia de 1/4 respecto a la longitud total (desde el centro hacia un extremo) hasta llegar al extremo de la pantalla (véase la Figura 3.3, donde el centro es B y un extremo puede ser A o C).

En las siguientes subsecciones, se detallará más a fondo el procedimiento que se realizó.

3.4.1. Cargar Archivo para Calibración

Al cargar el archivo, el botón con el nombre "Calibración (Máximos y Mínimos)" debe estar seleccionado; posteriormente, presione el botón "Buscar Archivo", como se muestra en la Figura 3.6.

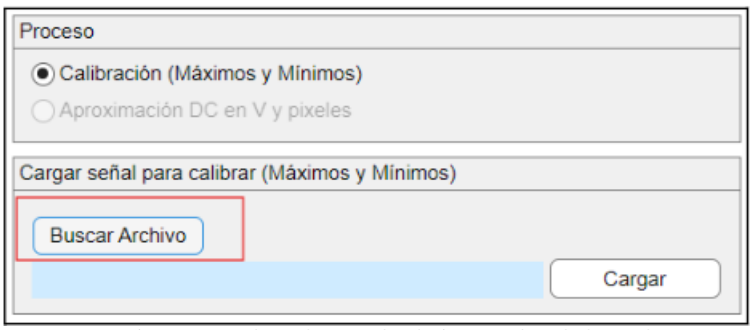


Figura 3.6: Botón para desplegar la búsqueda del archivo a utilizar.

Seleccionar el archivo en formato ".csv" a utilizar para normalizar la señal (en este caso el que lleva por nombre "dat1_0n0_05.csv" dentro de la carpeta "Calibración" dentro de la carpeta "Conjunto_senales") y obtener parámetros necesarios para reconstruirla; posteriormente, presione el botón de "Cargar" cuando vea la ruta del archivo

en el recuadro azul (véase la Figura 3.7 y 3.8).

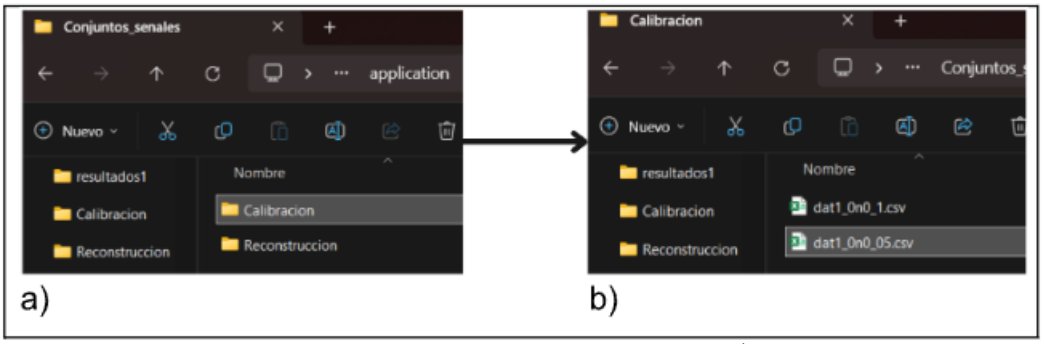


Figura 3.7: Archivo representativo de la señal a utilizar. a) carpeta con el archivo de calibración, b) archivo de calibración a utilizar.

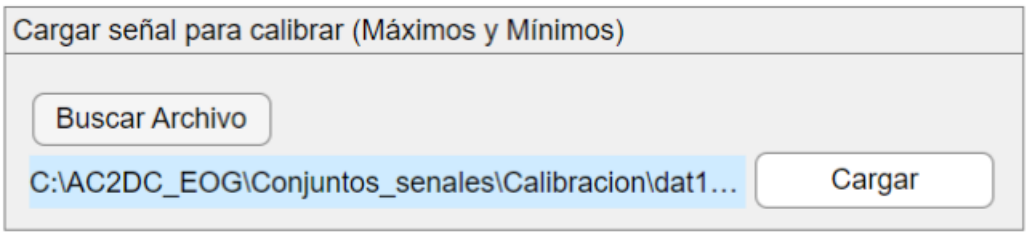


Figura 3.8: Botón para cargar el archivo seleccionado y comenzar a graficar la señal seleccionada.

3.4.2. Recortar y Suavizar Señal de Calibración Cargada

En el gráfico de la señal cruda, se debe seleccionar manualmente el valor que se desea utilizar de la señal para mantener la información de interés (ver Figura 3.9), posteriormente se debe definir manualmente el inicio y el fin de la señal, así como la cantidad de datos a emplear para la ventana de suavizado por media móvil (k) y la cantidad de datos para promediar y calcular el voltaje de offset desde el inicio hasta que transcurra la cantidad de datos establecida. Cuando estén definidos los parámetros previamente mencionados, presione el botón de "Recortar y suavizar" (véase la Figura 3.10).

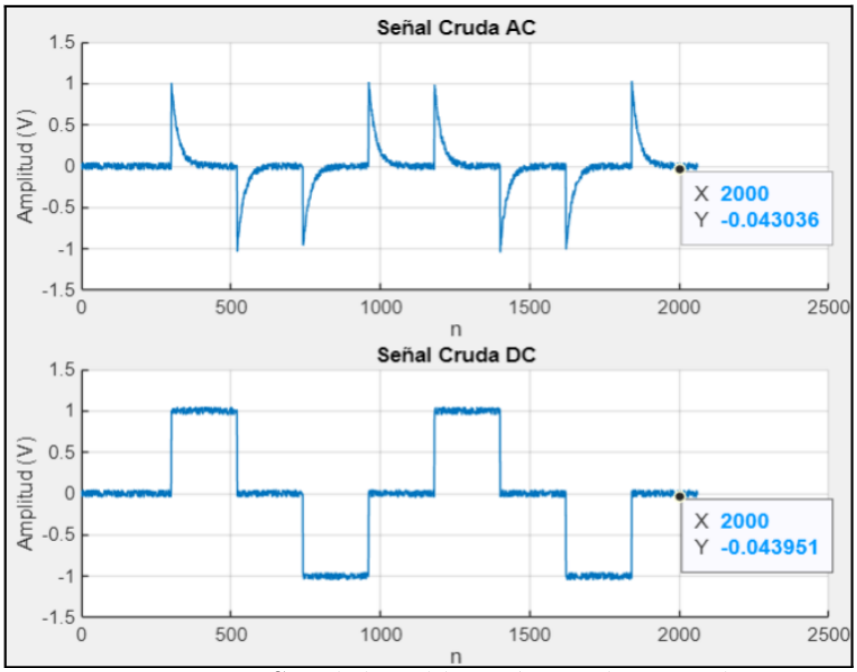


Figura 3.9: Señal de calibración cruda cargada.

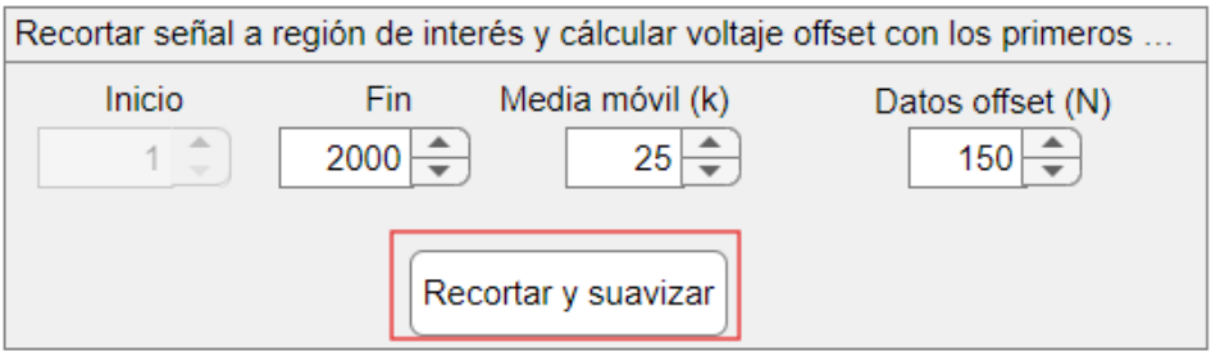


Figura 3.10: Parámetros para cortar y suavizar la señal cruda de calibración en el presente ejemplo.

Lo mostrado en la Figura 3.11 es el resultado de la señal al momento de haber presionado el botón de "Recortar y suavizar". En los gráficos se puede observar la señal en su forma "AC" demostrando como en el gráfico "Recortada" se aprecia la misma señal recortada a la cantidad de datos establecida; por otro lado, se observa en el gráfico de nombre "Recortada Suavizada" la señal suavizada por el filtro media móvil con los parámetros fijados en el paso anterior. Es importante mencionar que la señal se desplaza a un valor aproximado a cero dado que se calcula un valor de voltaje del offset obteniendo el promedio de la cantidad de datos (n) establecida en el control, el valor resultante es restado a cada dato contenido en la señal.

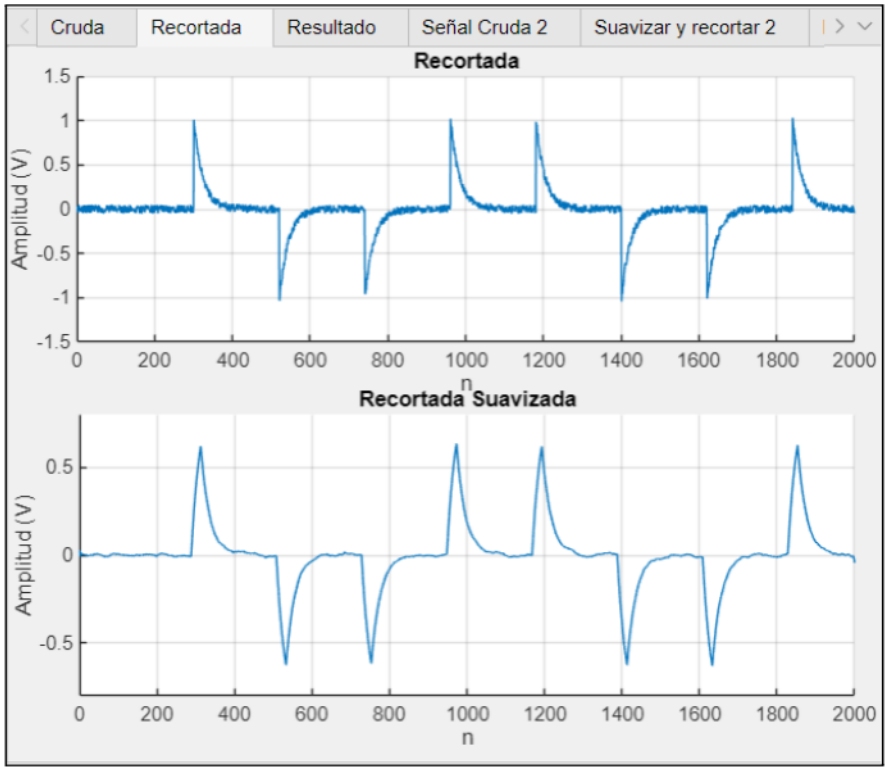


Figura 3.11: Gráfico representativo de la señal AC en su forma cruda pero recortada y suavizada.

3.4.3. Obtención de Datos Máximo y Mínimo Promedio

En este paso será necesario posicionar el cursor del mouse sobre puntos en donde se supere por primera vez el umbral hasta que esté por debajo de este nuevamente en cada sacádico generado por el movimiento ocular. En la Figura 3.12 se muestra la obtención manual de las distancias para la detección correcta de picos, tomando en cuenta la cantidad de datos transcurridos después de haber superado el umbral deseado para la detección hasta estar por debajo de dicho umbral nuevamente.

Figura 3.12: Obtención manual de las distancias para detección correcta de picos.

Teniendo en cuenta el paso anterior, será buscada la diferencia entre los índices establecidos por el usuario en los controles que se observan en el bloque de la Figura 3.13 (capturar Umbral y Distancia entre datos para nueva detección); posteriormente, presionar el botón con el nombre "Resultado" sin marcar la casilla "Promediar picos mínimos".

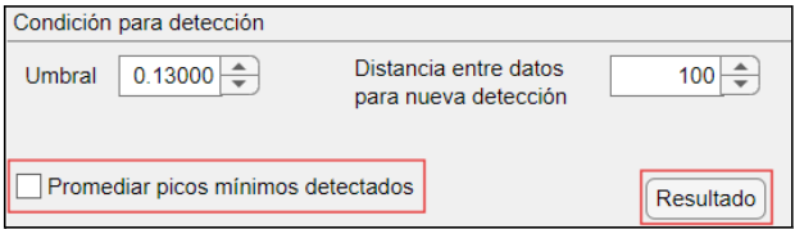


Figura 3.13: Sección para definir el umbral de detección picos y cantidad de datos a esperar para una nueva detección una vez que se superó el umbral.

Véase que en la Figura 3.14 se graficaron los picos correspondientes a la señal de calibración y en la Figura 3.15 se muestra el resultado del promedio de los picos máximos y mínimos en dicha señal. El resultado será útil para la conversión del valor de la amplitud en voltaje de la señal que se usará para aproximar a DC a valores que correspondan a coordenadas en píxeles de una pantalla con la resolución mostrada en la parte superior derecha de la Figura 3.2.

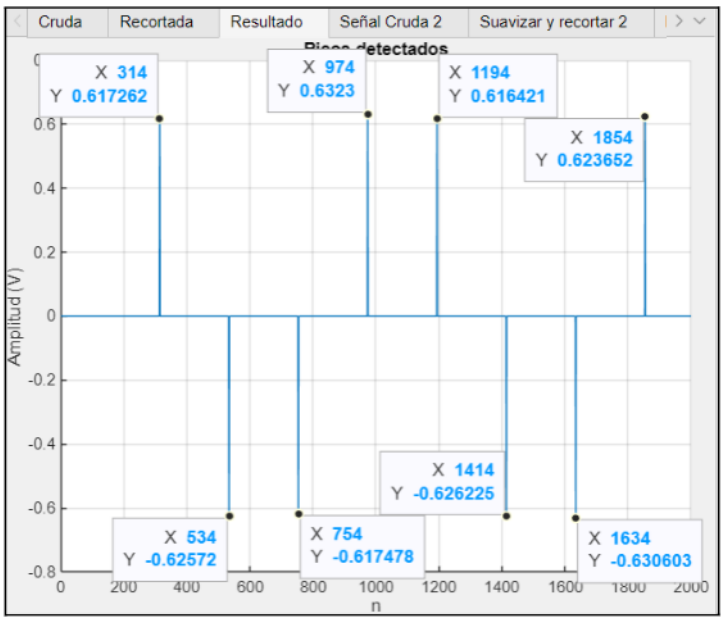


Figura 3.14: Picos detectados en la señal de calibración con el nombre de "dat1_0n0_05.csv".

Figura 3.15: Resultado del promedio de los picos máximos y mínimos obtenidos de la señal AC de calibración.

3.4.4. Carga de Archivo para la Reconstrucción DC

Dentro del modo 1, se cargará una señal a evaluar, lo cual quiere decir que se va a reconstruir una señal a la de DC utilizando los picos detectados en AC del movimiento ocular, además para aproximar a pixeles se utilizarán los picos máximos y mínimos calculados anteriormente.

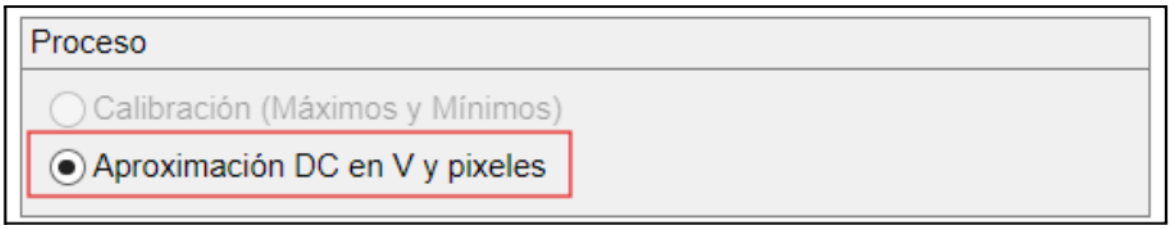


Figura 3.16: Selección de la señal para la reconstrucción de una señal aproximada a la de DC.

Lo siguiente es regresar al bloque de la Figura 3.6 en donde se repetirá el procedimiento de búsqueda de un archivo como se hizo en la sección 3.4.1 “Cargar Archivo para Calibración”, en este caso se seleccionará y cargará el archivo que contenga la señal AC que se utilizará para la aproximación a la señal DC. El archivo a seleccionar es el que se muestra marcado en la Figura 3.17 b).

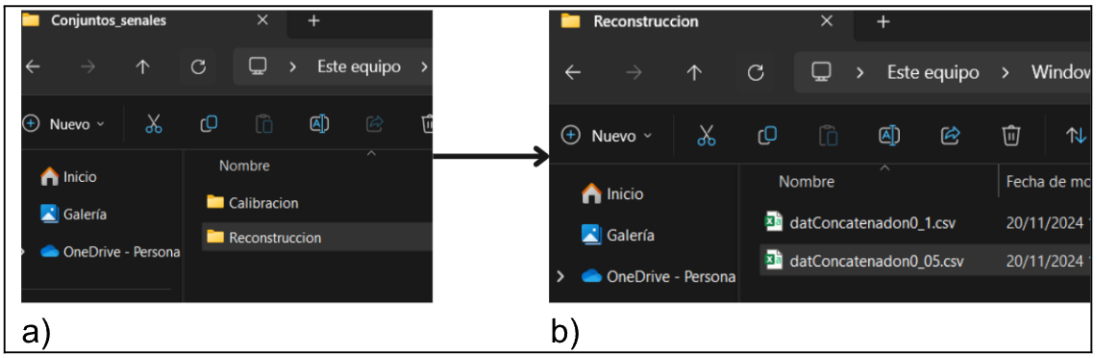


Figura 3.17: a) Carpeta a seleccionar en donde se encuentra el archivo a cargar. b) Archivo de señales DC y AC con movimientos más complejos seleccionar que se utilizará para aproximación a su forma en DC.

3.4.5. Recortar y Suavizar la Señal a Utilizar para Reconstrucción

A continuación, será necesario analizar la señal manualmente para determinar hasta qué dato se deberá recortar la señal, como ejemplo, en la Figura 3.18 se observa que la longitud que se establecerá será de 10200 datos porque el resto de la señal no aporta información relevante, por lo tanto el control "Fin" mostrado en la Figura 3.19, deberá tener el valor de 10200.

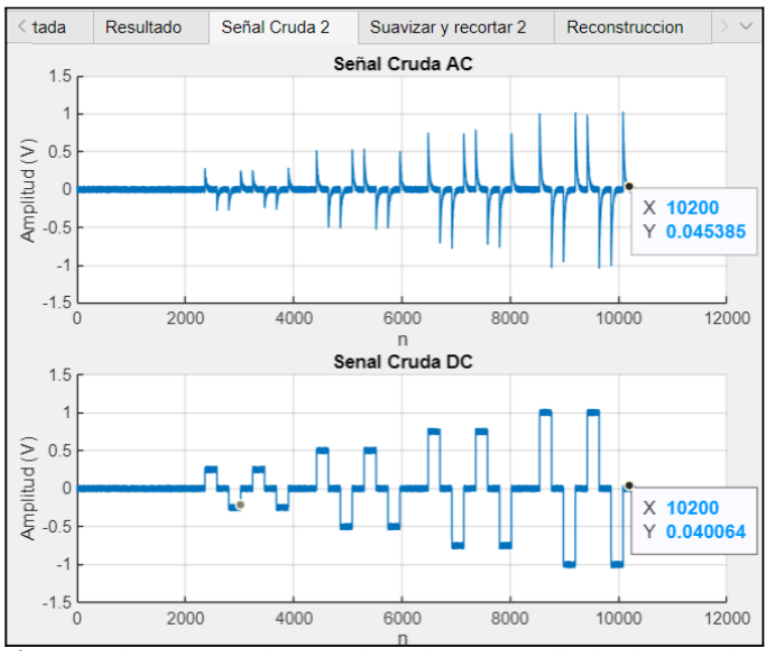


Figura 3.18: Información contenida en el archivo ".csv" de la señal utilizada para la estimación a su parte en DC.

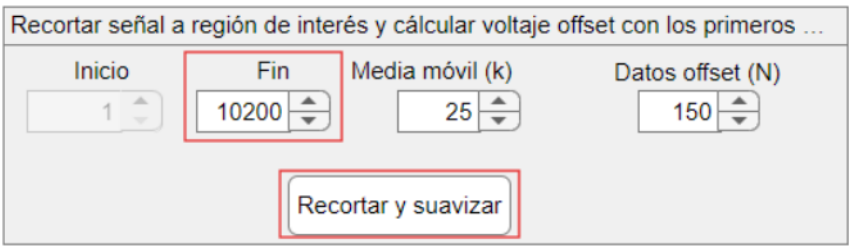


Figura 3.19: Parámetros para cortar y suavizar la señal cruda que será utilizada para la aproximación en DC.

El resto de controles deberá permanecer igual que cuando se suavizó y recortó la señal para calibrar en la sección 3.4.2 "Recortar y Suavizar Señal de Calibración Cargada". El resultado es el que se observa en la Figura 3.20; en este caso, hay que asegurarse

de que el valor del umbral no sea superado por ruido en algún lugar del arreglo de datos.

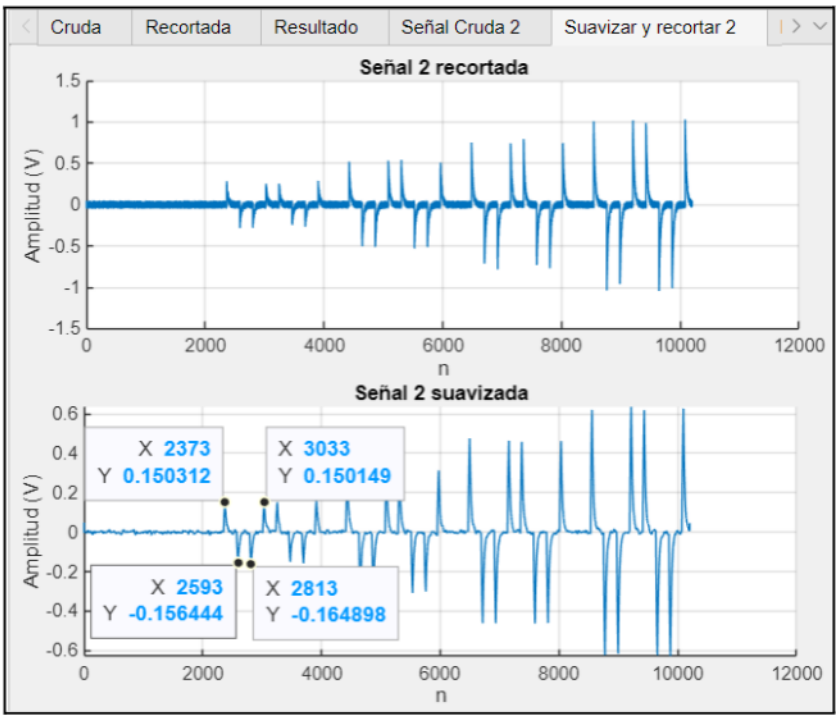


Figura 3.20: Señal a utilizar para aproximación a DC recortada y suavizada.

3.4.6. Calcular Aproximación a Señal DC utilizando la Señal AC

Para el cálculo de la aproximación a DC es necesario mantener los parámetros como se muestran en la Figura 3.13 y presionar el botón de "Resultado" y espere a que termine el procesamiento, puede durar unos segundos. Finalmente, en la Reconstrucción, el cual se aprecia en la Figura 3.21, existen dos señales parecidas a la señal en DC sin el problema de la deriva constante de la señal. Cabe destacar que en la Figura 3.21 hay dos gráficos, la primera de estas representa el valor de voltaje en la señal reconstruida y la otra muestra la aproximación en coordenadas en píxeles.

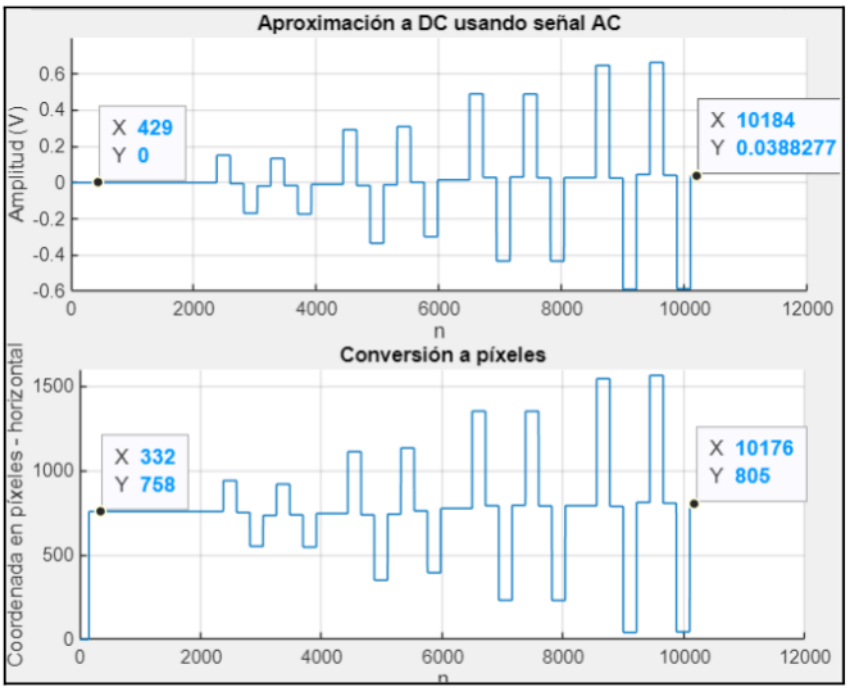


Figura 3.21: Aproximación a la señal en su forma DC en voltaje y píxeles utilizando la parte en AC.

En caso de querer guardar los gráficos de la Figura 3.21 en modo de arreglos en un archivo ".csv", será necesario asignar un nombre a la casilla de "Nombre de archivo"mostrada en la Figura 3.22 y presionar el botón guardar. El nombre asignado será el nombre del archivo ".csv" creado.

Figura 3.22: Sección para guardar los arreglos obtenidos de aproximación a DC.

3.5. Segundo Modo de Funcionamiento

En este modo se utilizaron dos controles para establecer umbrales, el primer control de umbral (mismo que se utilizó en la sección de "calibración") fue utilizado para determinar los valores de voltaje Máximos y Mínimos que son equivalentes al extremo derecho e izquierdo del campo de visión, y el segundo control fue utilizado para determinar el umbral de picos mínimos detectados (U.P.M.), el cual será utilizado para permitir la detección de picos al superar dicho umbral al momento de realizar la aproximación de AC a DC, así como descartar señales de pequeños movimientos oculares involuntarios,

en un nivel de sensibilidad no adecuado para registrarlos. Este modo de funcionamiento es aplicable en la simulación en donde la señal almacenada contiene la información de calibración junto a la información que será utilizada para la aproximación a DC.

En este modo se utiliza el primer valor de umbral para calibración (Máximos y Mínimos) y el segundo control que determina el umbral de picos mínimos detectados (U.P.M.) será utilizado para permitir la detección de picos al superar dicho umbral al momento de realizar la aproximación de AC a DC, así como descartar señales de pequeños movimientos oculares involuntarios, en un nivel de sensibilidad no óptimo para registrarlos. Este modo de funcionamiento es aplicable en la simulación en donde la señal almacenada contiene la información de calibración junto a la información que será utilizada para la aproximación a DC.

3.5.1. Obtención de Datos para la Normalización

Al igual que en la Sección 3.4 "Primer Modo de Funcionamiento", se procederá a describir una serie de pasos en los que el objetivo es obtener valores correspondientes al movimiento ocular del centro al extremo de un campo de visión para determinar máximos y mínimos; en este caso, se cargará una señal que contenga los máximos y mínimos, pero sin ser exclusivos, debido a que en dicha señal puede haber distintas amplitudes correspondientes a movimientos más pequeños en el movimiento ocular.

En este modo se utiliza el primer valor de umbral para calibración (Máximos y Mínimos) y el segundo control que determina el umbral de picos mínimos detectados (U.P.M.) será utilizado para permitir la detección de picos al superar dicho umbral al momento de realizar la aproximación de AC a DC, así como descartar señales de pequeños movimientos oculares involuntarios, en un nivel de sensibilidad no óptimo para registrarlos.

3.5.2. Cargar Señal para Obtener Datos Máximos y Mínimos

Para terminos y referencias ver Tabla A.1. El primer paso es cargar la señal a utilizar para calibración y estimación como se hizo en Sección 3.4.1 "Cargar Archivo

para Calibración", diferenciándose en que ahora se deberá presionar el botón buscar para posteriormente acceder a la carpeta llamada "Conjunto_senales_m2" y cargar la señal con el nombre de "IIRN_C1_P1-H_200Hz_G500_DCAC.csv" como se muestra en la Figura 3.23.

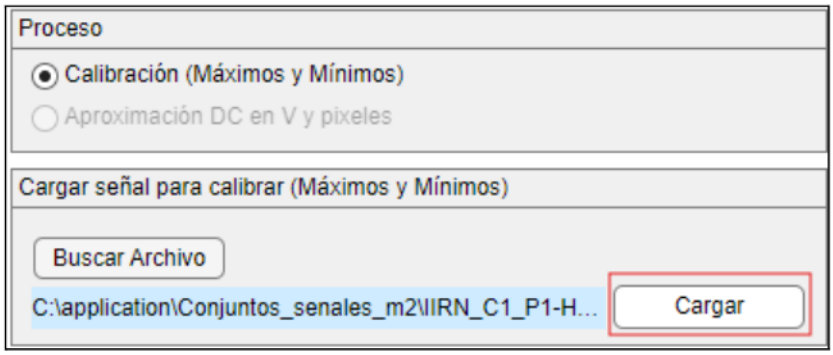


Figura 3.23: Carga de la señal a utilizar para el modo 2 de uso.

Lo obtenido como resultado son los gráficos de los arreglos que conforman al archivo ".csv", dicho archivo cargado está conformado por resultados de señal de EOG siguiendo cierta rutina tanto en su forma AC (parte superior) como en DC (parte inferior), véase la Figura 3.24.

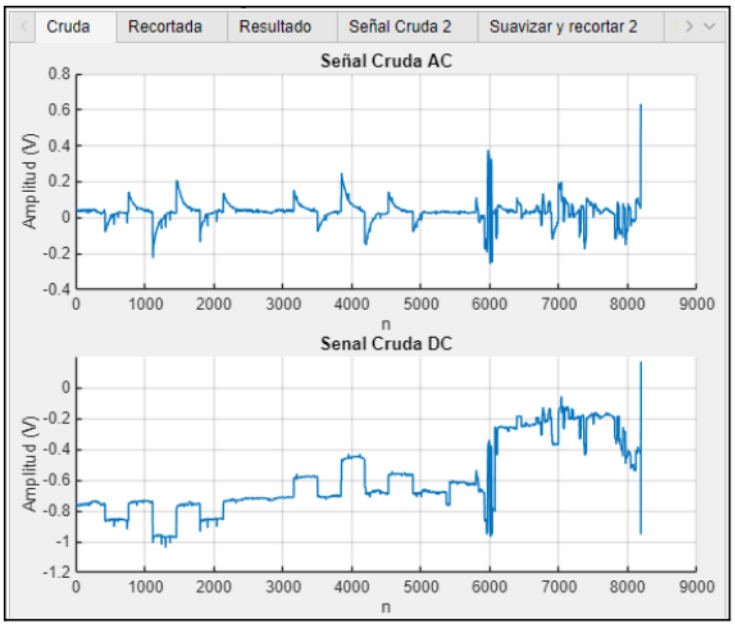


Figura 3.24: Señal EOG cargada en su forma de AC en la parte superior y en su forma de DC en la parte inferior.

3.5.3. Recortar y Suavizar la Señal

En este paso es necesario delimitar la señal para solamente trabajar con lo obtenido en la adquisición, como se puede observar en la Figura 3.25, manualmente se buscó el último dato permitido para trabajar.

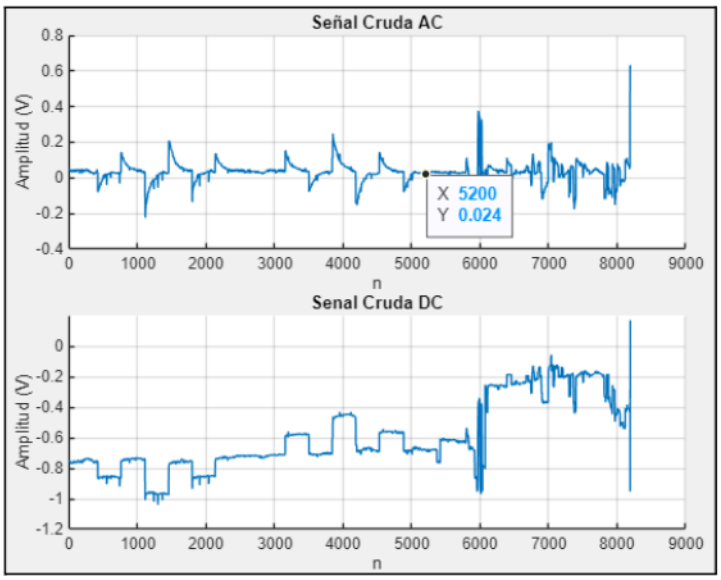


Figura 3.25: Punto en el que se desea cortar la señal, en este caso lo de interés es lo que se muestra desde el dato 1 hasta el 5200.

Una vez establecido en las gráficas cuál fue nuestro último dato de interés, será necesario configurar el bloque de la forma que se muestra en la Figura 3.26, una vez establecidos los valores, presionar el botón llamado "Recortar y suavizar", obteniendo unas gráficas suavizadas como se observa en la Figura 3.27.

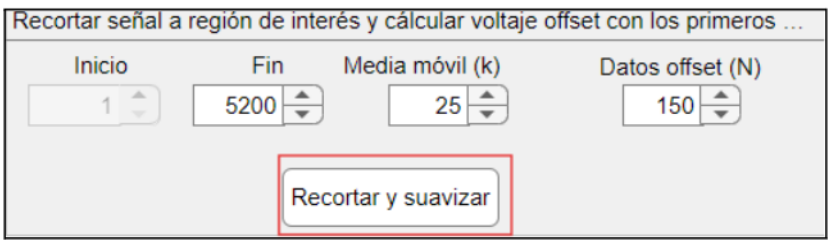


Figura 3.26: Parámetros para realizar el suavizado y recorte de la señal.

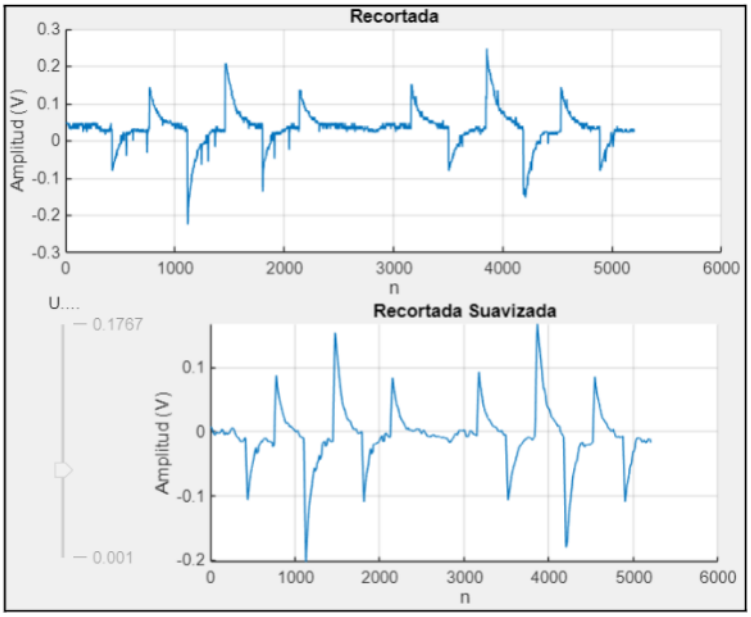


Figura 3.27: Resultado de la señal EOG en AC del archivo "IIRN_C1_P1-H_200Hz_G500_DCAC.csv" suavizada y recortada.

3.5.4. Obtención de Datos Máximo y Mínimo Promedio

Se establecerá una separación de las amplitudes correspondientes a los movimientos más grandes para la obtención de los datos máximos y mínimos. En la Figura 3.30 los picos más pequeños (que representan los movimientos más pequeños) fueron marcados para poder establecer un umbral por encima de ellos, así solamente serán detectados los picos más grandes de la señal. De igual manera, en la Figura 3.30, el pico más grande dentro del grupo de los más pequeños presentes es de -0.109268 (con valor absoluto sería 0.109268) por lo tanto, se puede llegar a establecer en el control "Umbral" un valor aproximado como de 0.115 dado que es un poco mayor a 0.109268 y menor a los valores del grupo de picos con mayor amplitud (véase la Figura 3.29). Para este modo se deberá marcar la casilla con la leyenda "Promediar picos mínimos detectados".

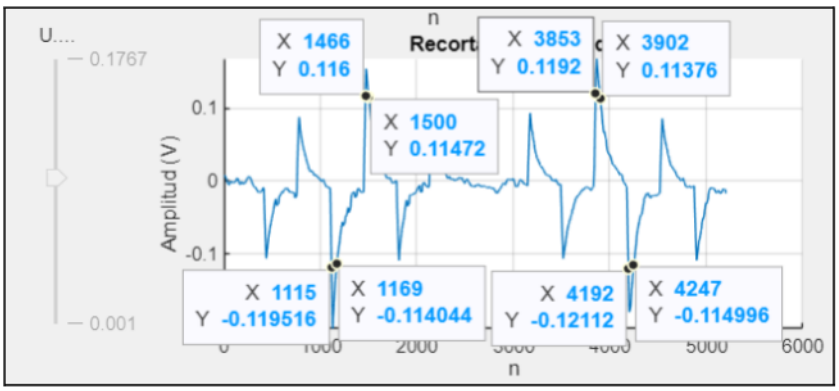


Figura 3.28: Marca de puntos en los movimientos sacádicos más pequeños.

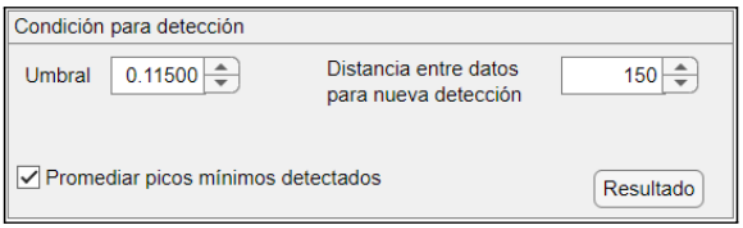


Figura 3.29: Establecimiento de valor de "Umbral" a 0.115.

Para establecer en el control "Distancia entre datos para nueva detección" se hace algo similar a lo establecido en "Umbral" pero en este caso, en el gráfico que se muestra en la Figura 3.30 se debe contar el número de datos que transcurren desde que se supera un umbral hasta que la señal cae por debajo de éste, tomando en cuenta los picos que corresponden a los desplazamientos más grandes en la señal.

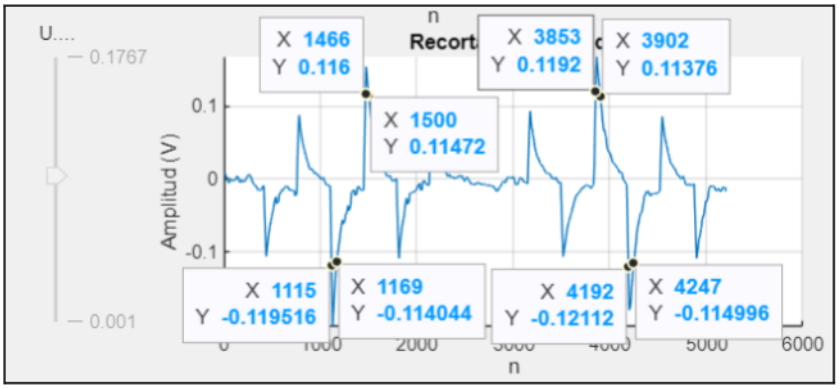


Figura 3.30: Estimación de distancia transcurrida entre la superación de "Umbral" hasta estar por debajo de dicho valor umbral.

Se aprecia que son menos de 100 datos para estar por debajo del umbral nuevamente, por lo que en el control "Distancia entre datos" se establecerá un valor de 100. Una vez se hayan asignado los valores como se ve en la Figura 3.31, se deberá presionar el botón con el nombre de "Resultado", espere a que termine el procesamiento, puede

durar unos segundos.

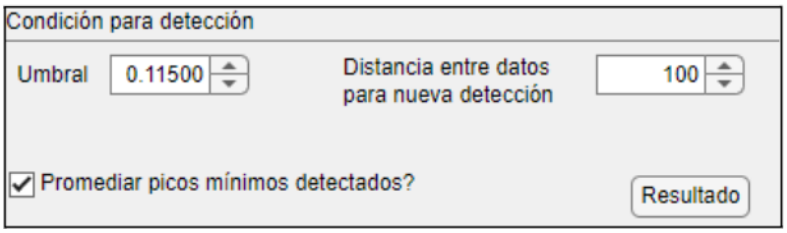


Figura 3.31: Valores en los distintos controles en el bloque "Condición para detección" para el modo dos del ejemplo presentado.

Lo obtenido en este proceso se muestra en la Figura 3.32, lo cual representa el valor promedio máximo obtenido al girar la vista del centro a un extremo, y como mínimo, el movimiento mayor hacia el otro extremo del campo de visión.

Figura 3.32: Datos máximos y mínimos obtenidos para el modo dos en el ejemplo presentado.

De igual manera, como resultado se obtuvo un arreglo de datos de la señal recortada y suavizada pero en su forma de valor absoluto, esto con la finalidad de poder usar el segundo umbral (slider mostrado en la Figura 3.33) para ajustar el umbral de detección de picos mínimos, con ello se podrían descartar picos más pequeños que posiblemente no sean necesarios o de interés en la reconstrucción.

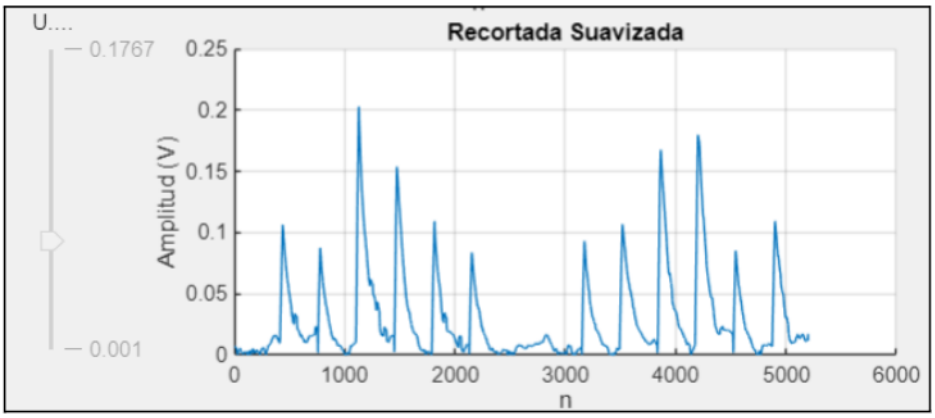


Figura 3.33: Señal suavizada, suavizada y en valor absoluto.

3.5.5. Ajustar el Umbral de Detección para Picos Pequeños de Interés

El propósito es establecer un valor de umbral (Figura 3.34) para el Slider llamado "U.P.M." de manera que sea inferior al pico más pequeño de interés de valor absoluto. En la Figura 3.34 se observan dos recuadros, el rojo muestra el valor del pico más pequeño en la señal y el verde el valor del Slider. En caso de querer descartar ciertos picos se requerirá obtener manualmente (buscando con el cursor en la gráfica) el valor de cada pico y de esa forma establecer un umbral específico.

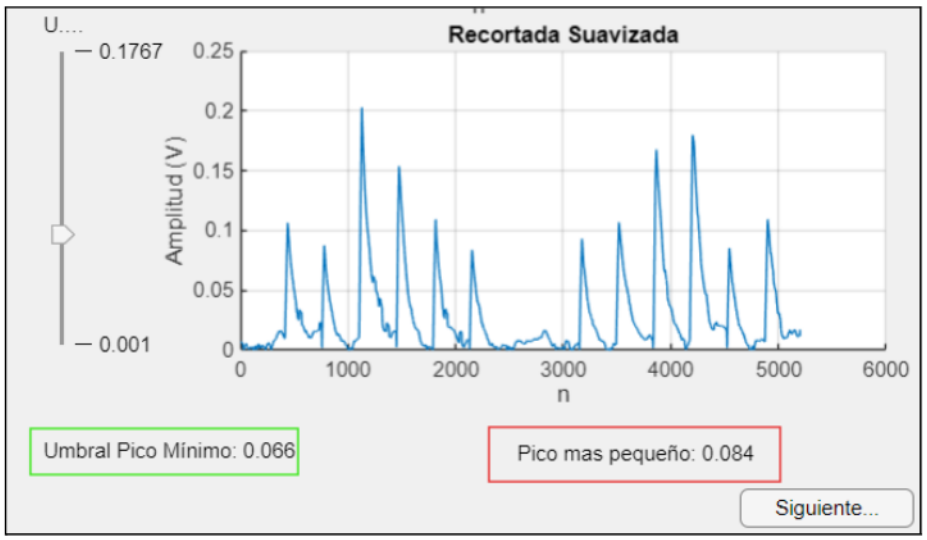


Figura 3.34: Ajuste del Slider con el nombre "pos" a un valor de umbral permitido.

Cuando se establezca este valor de umbral se deberá presionar el botón "Siguiente" del gráfico de la Figura 3.34. Como resultado, el botón de siguiente quedará inhabilitado y será posible continuar con el proceso.

3.5.6. Reconstrucción de Señal Aproximada a DC

Se estimará una señal correspondiente a DC a partir de las amplitudes de la señal AC descartando los picos que no superen el umbral "U.P.M" establecido en el paso 3.5.5 "Ajustar el umbral de detección para picos pequeños de interés".

3.5.7. Cargar Señal a Aproximar a DC

Deberá ser cargada la misma señal que en el paso 3.5.2 “Cargar Señal para Obtener Datos Máximos y Mínimos” seleccionando esta vez “Aproximación DC en V y pixeles”, como se muestra en la Figura 3.35.

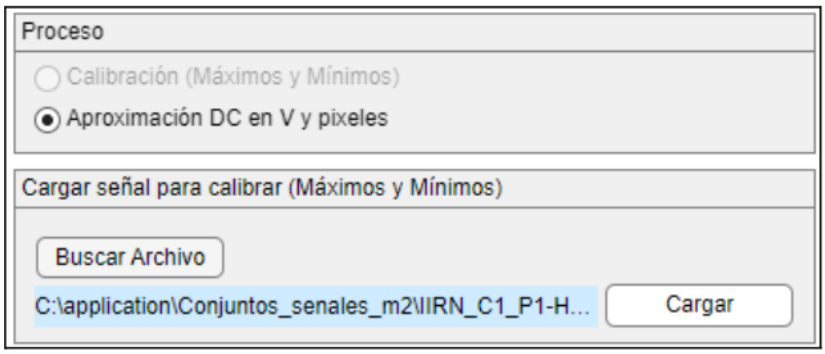


Figura 3.35: Configuración de carga de archivo “IIRN_C1_P1-H_200Hz_G500_DCAC.csv” configurando el proceso para aproximar resultado en DC.

3.5.8. Recorte y Suavizado de la Señal

Para la parte del recorte y suavizado de señal, lo recomendable es dejar la configuración previamente establecida en 3.5.2 “Recortar y suavizar la señal” (véanse las Figuras 3.25, 3.26 y 3.27).

3.5.9. Obtención de Resultados

El bloque “Condición para detección” permanece igual que lo establecido en el apartado 3.5.4 “Obtención de Datos Máximo y Mínimo Promedio”, en esta ocasión solamente se presiona el botón de resultado como se muestra en la Figura 3.36, espere a que concluya el procesamiento; puede durar unos segundos.

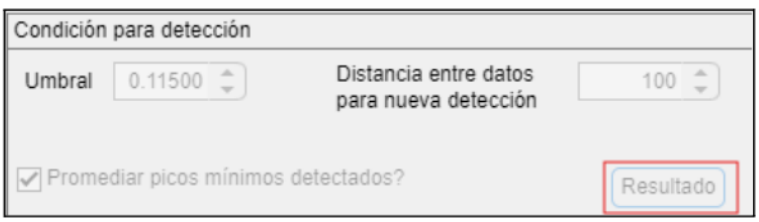


Figura 3.36: Configuración de umbral y nueva detección en etapa de reconstrucción a aproximación DC.

Finalmente, se obtiene como resultado en “Reconstrucción”lo mostrado en la Figura 3.37, en donde se observan dos gráficos, el superior representa los posicionamientos de la vista con amplitud en voltaje y la gráfica inferior está representado por un valor en píxeles o coordenada en base a la resolución del campo de visión. Se añadió una opción para almacenar los arreglos resultantes en un archivo “.csv” asignando solamente un nombre al archivo y presionando el botón “guardar”.

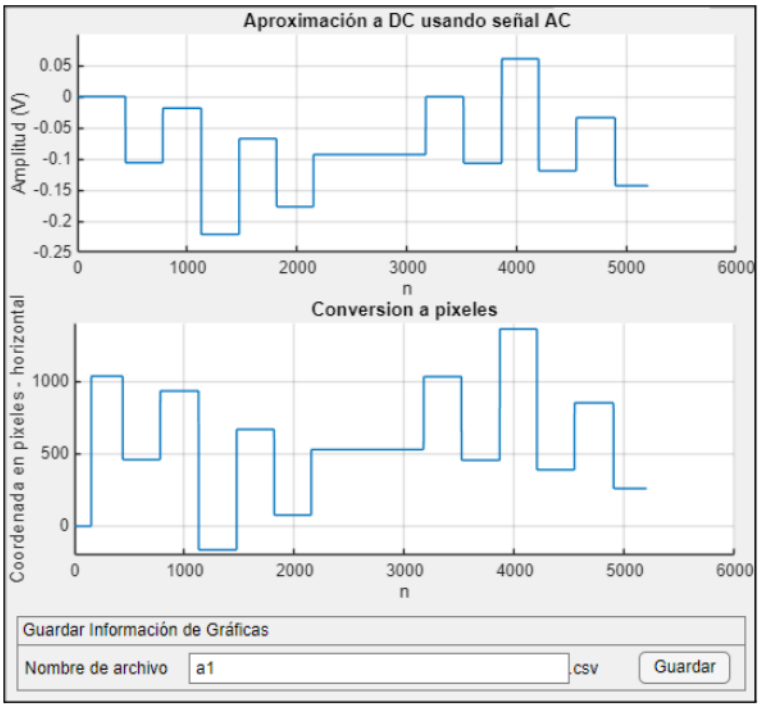


Figura 3.37: Señales resultantes del procedimiento de aproximación a DC en base a señales EOG en AC.

Interfaz

Se desarrolló un programa de cómputo que simula la adquisición de señales electrooculografícas (EOG) en tiempo real, para la evaluación de algoritmos de procesamiento de señales EOG provenientes de desplazamientos oculares sacádicos en horizontal, a píxeles que representan dimensiones de desplazamiento dentro de un campo de visión definido. Cuenta con una interfaz que permite calibrar y evaluar señales EOG en AC (señales EOG DC que fueron tratadas con un filtro pasa ALTA), para reconstruir las correspondientes señales EOG en DC sin problemas de deriva ni ruido, para su interpretación en desplazamientos.

4.1. Instalación de la aplicación del programa de cómputo para el usuario

Los pasos a seguir son los siguientes:

1. Descargar la siguiente carpeta: AC2DC_EOG.

2. Abrir la carpeta que contiene el archivo de instalación (véase Figura 4.1).

Figura 4.1: Carpeta con archivo de instalación.

3. Ejecutar el archivo ".exe" llamado "Instalador.exe" (véase Figura 4.2).

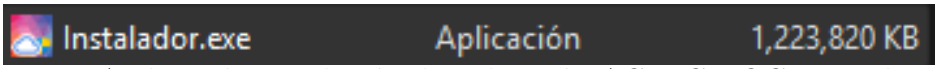


Figura 4.2: Archivo de instalación de aplicación AC2DC_EOG para el usuario.

4. En la ventana emergente, hacer click en el botón "siguiente" (véase Figura 4.3).

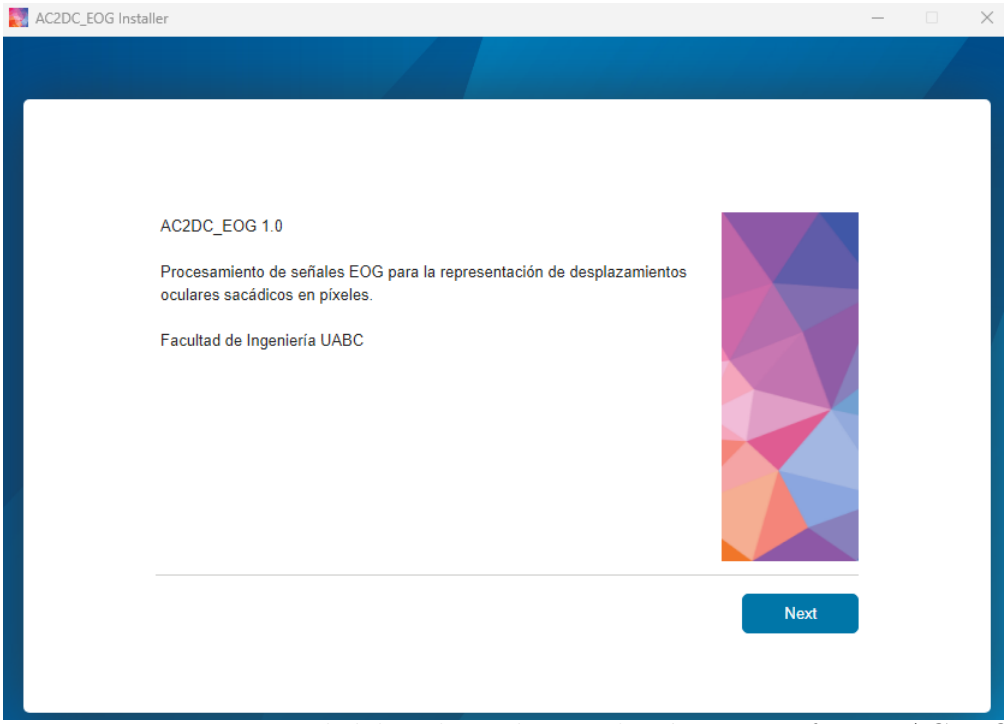


Figura 4.3: Ventana principal del archivo de instalación para software AC2DC_EOG (Procesamiento de señales EOG para la representación de desplazamientos sacádicos en píxeles).

5. A continuación, es necesario cambiar la carpeta de destino de instalación como se muestra en la siguiente imagen, dejando el recuadro como "C:AC2DC_EOG" para evitar problemas al momento de ejecutar el programa. Si lo desea, de manera opcional puede agregar un atajo al escritorio marcando la check box que aparece en la ventana (veáse Figura 4.4.

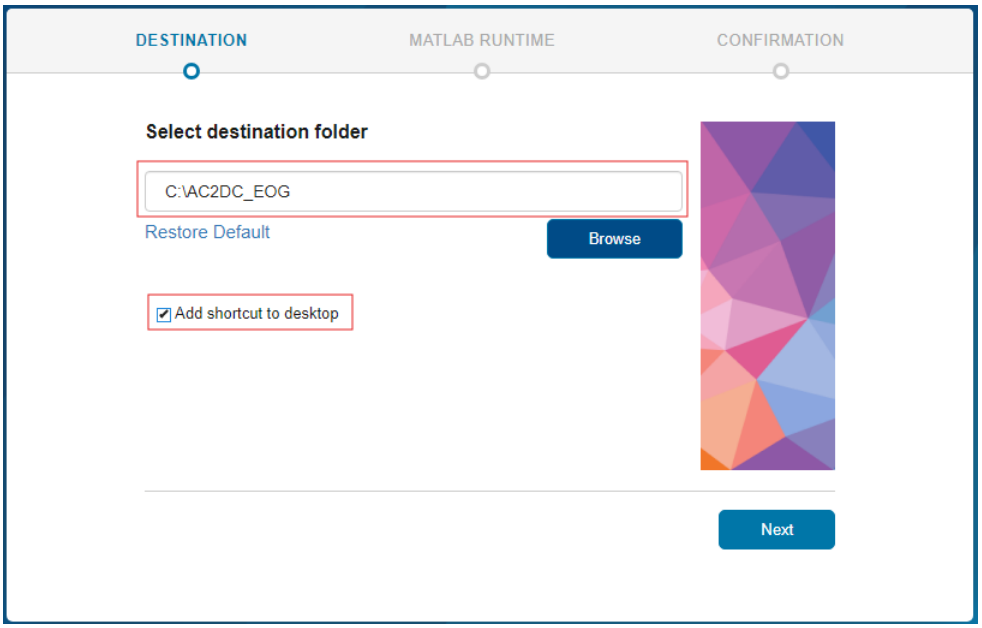


Figura 4.4: Pantalla de instalación del archivo ejecutable.

6. En "MATLAB runtime" dar click en "siguiente" sin cambiar el folder de destino (véase la Figura 4.5).

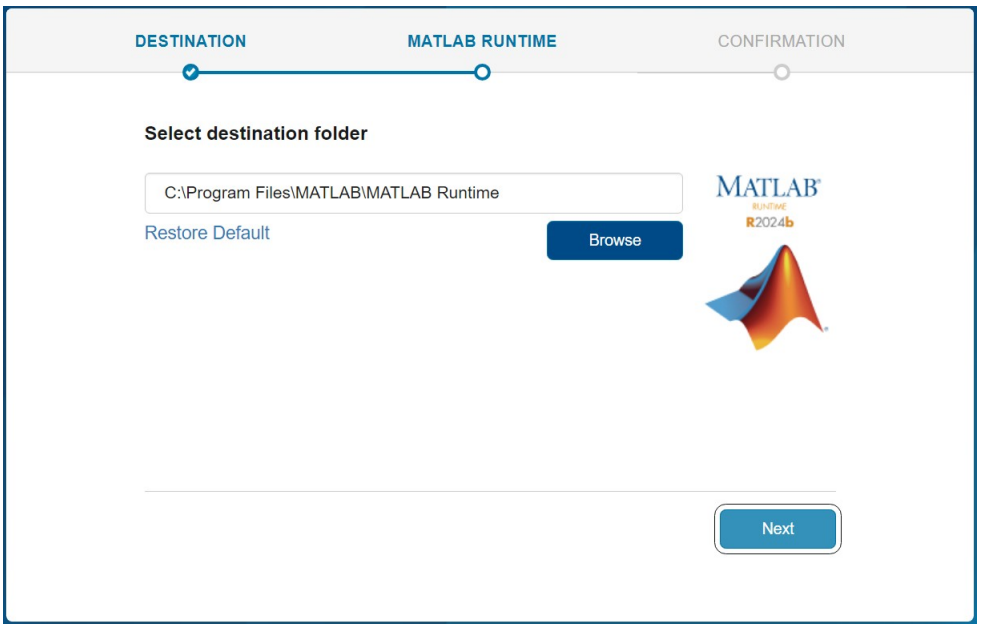


Figura 4.5: Pantalla de instalación de librerías a utilizar.

7. La siguiente pantalla es para aceptar términos y condiciones, aquí se deberá marcar "sí" y posteriormente presionar el botón "siguiente" (véase la Figura 4.6).

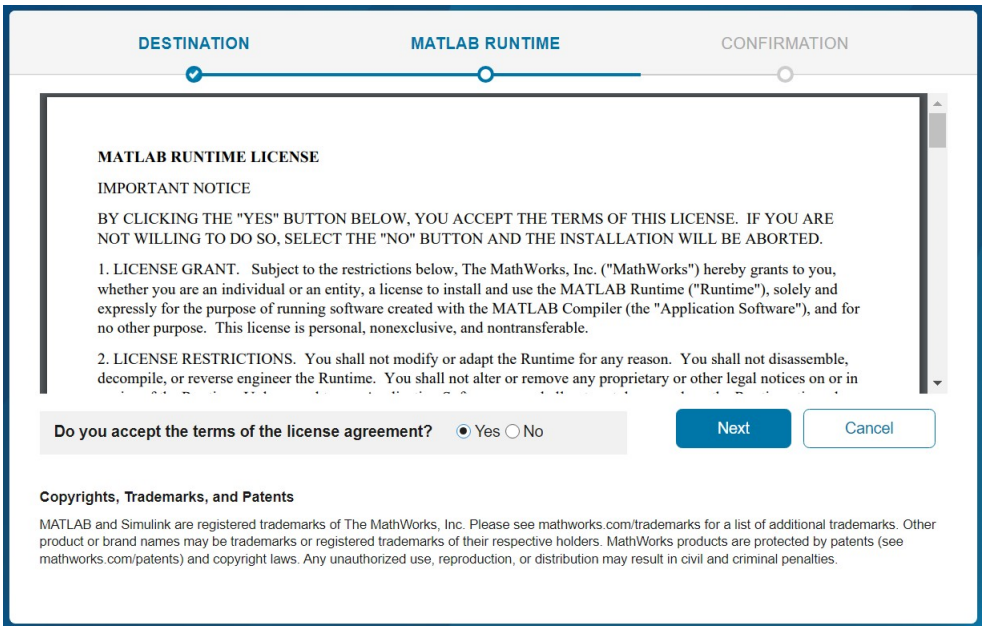


Figura 4.6: Pantalla de términos y condiciones del software MATLAB.

8. Presione el botón “Comenzar instalación” y espere (véase la Figura 4.7).

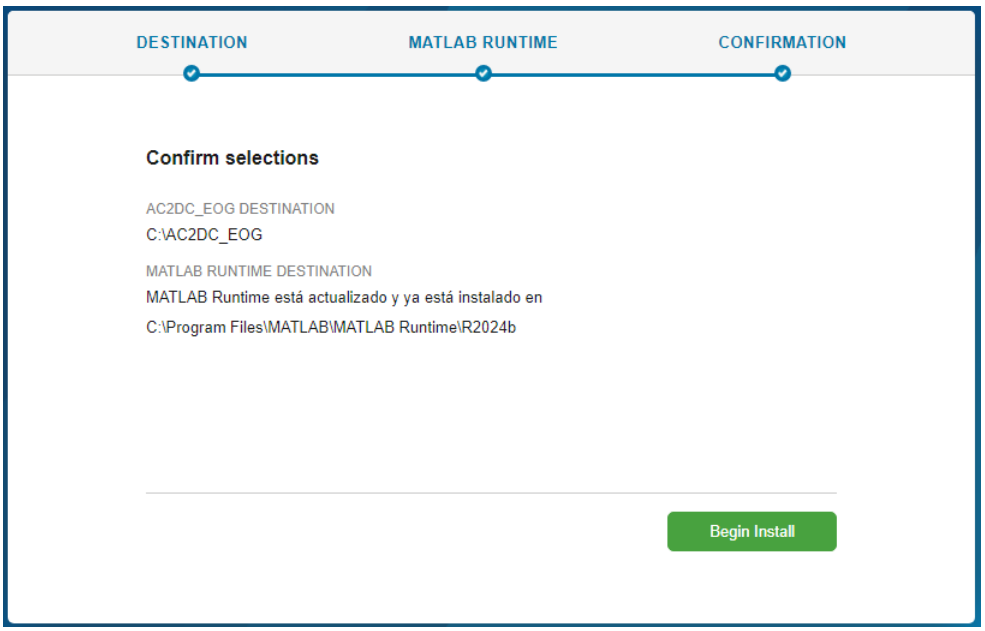


Figura 4.7: Pantalla de confirmación de la instalación.

9. Una vez se haya terminado la instalación, deberá presionar el botón que dice cerrar (véase la Figura 4.8).

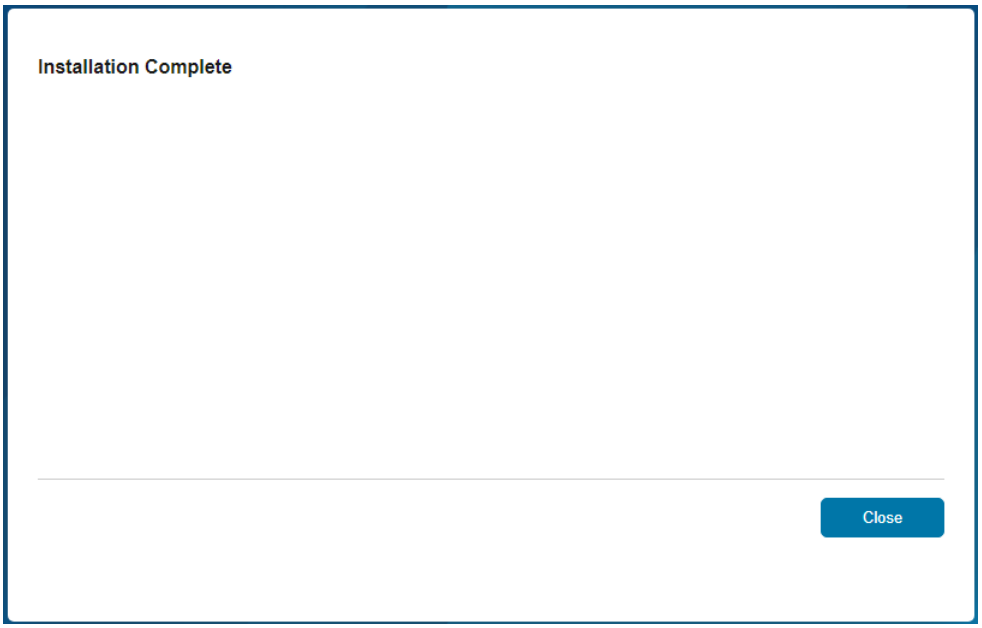


Figura 4.8: Pantalla mostrando instalación completa.

Siguiendo los pasos descritos hasta este punto es posible abrir el archivo ejecutable AC2DC_EOG.exe, (ver Figura 4.9) que se encuentra en la ruta donde fue instalada la aplicación: `C:\AC2DC_EOG\application`.

10. En la carpeta compartida con el nombre "AC2DC_EOG", misma que contiene al archivo de instalación, deberá copiar la carpeta con el nombre "Conjunto_senales" y "Conjunto_senales_m2" y pegarla en la carpeta creada al momento de la instalación con él, específicamente en la ruta `C:\AC2DC_EOG\application`. Con lo anterior, deberá ser posible ejecutar el programa sin complicaciones.

11. Abrir el programa haciendo doble click en el acceso directo creado en el escritorio en caso de haber marcado la casilla en el paso 5 de la presente sección. En el caso de no haber marcado la casilla, es posible acceder al programa siguiendo la ruta `C:\AC2DC_EOG\application` y dar doble click sobre el archivo llamado "AC2DC_EOG.exe" (véase la Figura 4.9).

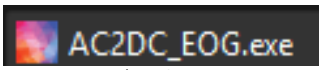


Figura 4.9: Archivo ejecutable.

4.2. Desinstalación de la aplicación del programa de cómputo para el usuario

Para desinstalar la aplicación del equipo hay dos pasos posibles a seguir:

Método 1 de desinstalación:

1. Ubicarse en la ruta `C:\AC2DC_EOG\uninstall\bin\win64` y busqué el archivo con el nombre "Uninstall_Application.exe" (véase la Figura 4.10).

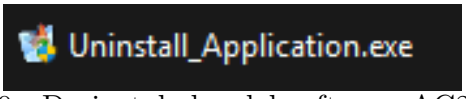


Figura 4.10: Desinstalador del software AC2DC_EOG.

2. Se desplegará una ventana para confirmar la desinstalación, hacer click en "sí".

3. Esperar a que se complete la desinstalación.

Método 2 de desinstalación:

1. Ir a la configuración de windows y siga la ruta "Aplicaciones ≫ Aplicaciones instaladas" y busque AC2DC_EOG (la aplicación instalada).

2. Presione en los tres puntitos mostrados a la derecha de la aplicación y a continuación, presione desinstalar (véase la Figura 4.11).

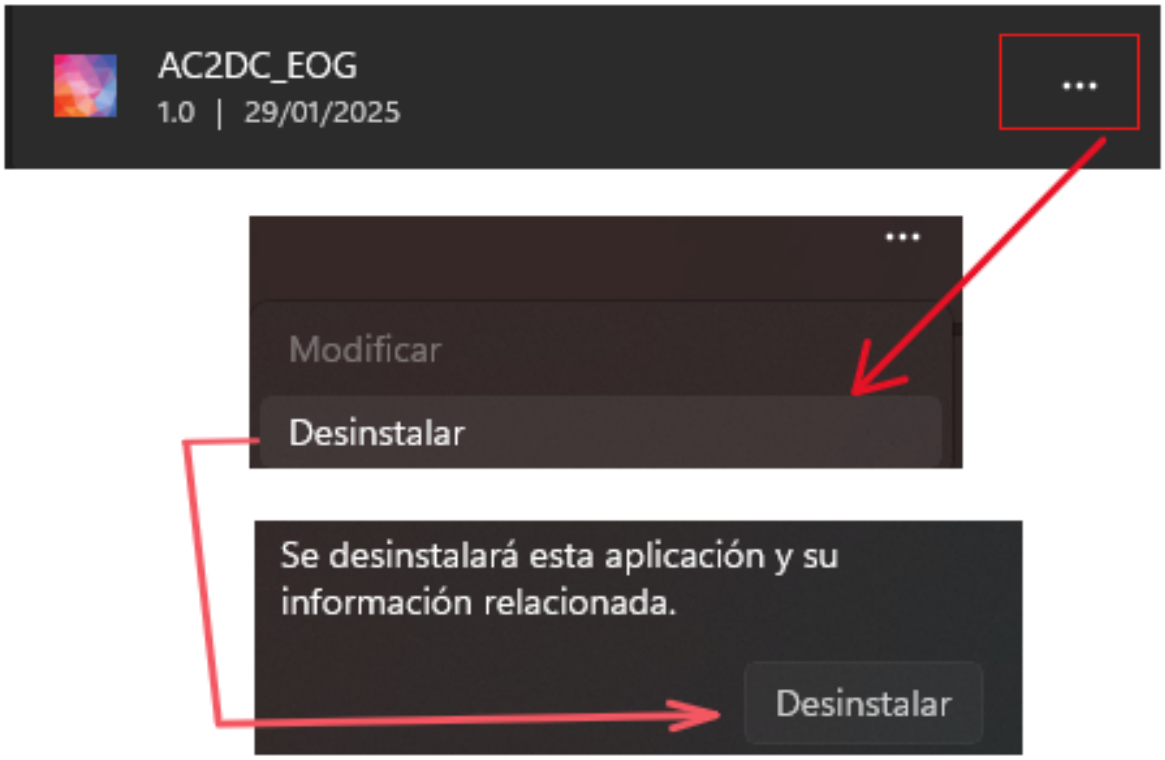


Figura 4.11: Desinstalación de software AC2DC_EOG

3. En la ventana emergente, dar click en "Sí" como se muestra en la Figura 4.12.

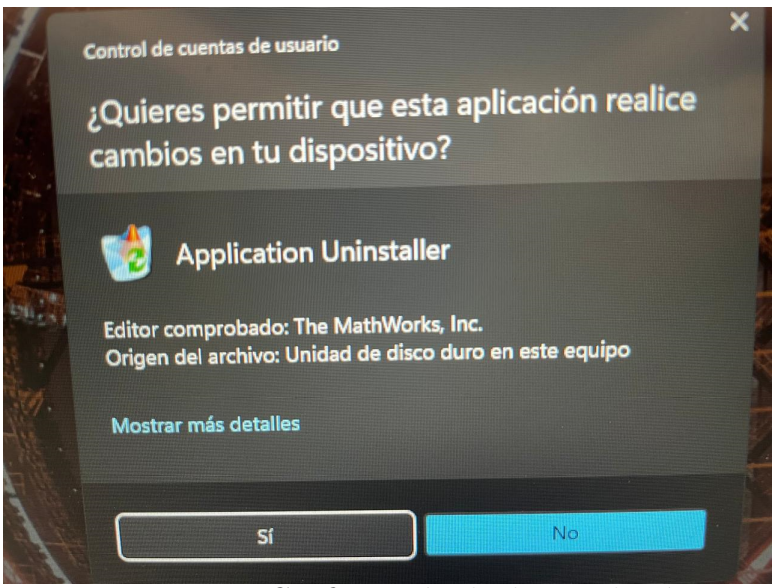


Figura 4.12: Confirmación de desinstalación

4. Esperar a que se complete la desinstalación (véase la Figura 4.13).

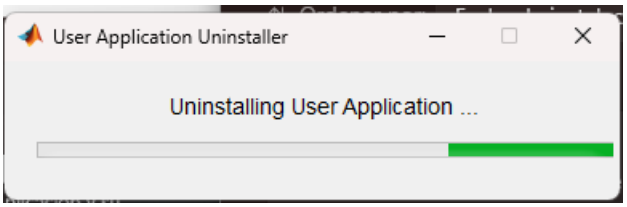


Figura 4.13: Pantalla de desinstalación de la aplicación.

4.3. Instalación del entorno de programación para el desarrollador

Requerimientos, contar con sistema operativo Windows y tener previamente instalado el software de MATLAB con los toolbox:

- Signal Processing Toolbox.
- Curve Fitting Toolbox.

Los pasos a seguir para tener acceso al algoritmo son los siguientes:

1. Descargar la siguiente carpeta: Desarrollador.
2. Descomprimir la carpeta descargada y colocarla en la ruta C:
3. Inicie el software MATLAB (con la versión disponible).

Figura 4.14: Software MATLAB versión R2024b.

4. Para ingresar al entorno del desarrollador, abra el archivo de código principal mainx.m presionando el icono “Open”, y buscando en la ruta el archivo, finalmente presione “Abrir”.

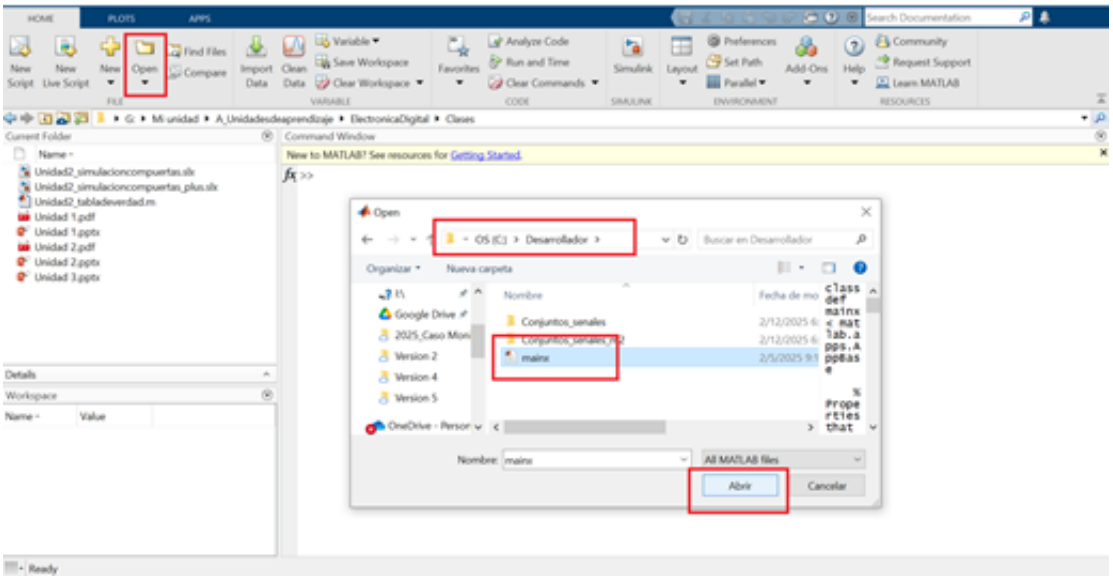


Figura 4.15: Apertura de archivo con código de programación mainx.m en el entorno de MATLAB.

5. A continuación se explica la función de los bloques de código destacables del programa.

a) Se define la clase para encapsular datos (propiedades) y funciones (métodos) en una única estructura lógica, facilitando la reutilización y organización del código. Ver Apéndice B.

b) Inicializar variables a utilizar dentro de la app. Ver Apéndice C.

c) Función encargada de la carga del conjunto de señales, la cual se activa al presionar un botón en el panel frontal, este lleva por nombre “Cargar”. Un ejemplo de la carga de archivos se muestra en las Figuras 3.6, 3.7 y 3.8). Ver Apéndice D.

d) En la siguiente función se hace un suavizado por media móvil sobre la señal, además se recorta la señal desde el inicio hasta donde se establece el valor de “Fin” (véase la Figura 3.11). Ver Apéndice E.

e) Función encargada de dos propósitos. Primero, para la etapa de calibración en la cual se calculan los picos de la señal AC suavizada, con base en los parámetros establecidos en la presente etapa. Segundo, para realizar la aproximación a señal DC utilizando la señal AC del EOG. A continuación se muestra el código de dicha función. Ver Apéndice F.

f) Esta función es útil para definir qué valor se puede considerar como umbral para la

detección de picos. La función es usada para la etapa de reconstrucción o aproximación de la señal DC y no en la de calibración. Ver Apéndice G.

g) Función creada para cambiar propiedades de los controles en el Tab de "Recortada", al presionar click en el botón de "Siguiente", este se va a desactivar al igual que el slider mostrado en esa pestaña. Ver Apéndice H.

h) El siguiente código describe el procedimiento para almacenar lo mostrado en el Tab "Reconstrucción" (como se muestra en la Figura 3.21) en modo de arreglos de 2 columnas, en la primera columna se almacenan los valores de lo reconstruido en voltaje mientras que en la segunda columna es almacenado el valor en coordenadas tomando en cuenta la resolución del campo de visión. Ver Apéndice I.

i) Proceso para regresar al estado inicial conservando la información de calibración, dicha información es la que se obtiene en el sección 3.4.3 del primer modo de funcionamiento. Ver Apéndice J.

j) La siguiente función se usa para limpiar variables, de este modo se regresan a un estado inicial indicando a los arreglos estar vacíos y las variables numéricas a estar inicializadas con un valor de 0. La actual función no reinicia o elimina los valores obtenidos en la etapa de calibración, los cuales son valor máximo y mínimo. Ver Apéndice K.

k) Esta sección de código es encargada de eliminar los valores calculados para la calibración, los cuales son el valor máximo y el valor mínimo promedio. También se restablece el color del indicador lámpara a rojo, indicando que fueron restablecidas las variables anteriormente mencionadas. Ver Apéndice L.

l) Función encargada de limpiar todos los gráficos incluidos en el Tab Group. De igual forma, el siguiente código indica que se muestra en cada gráfico utilizado. Ver Apéndice M.

m) En el siguiente bloque de código son implementadas funciones para realizar la impresión de valores calculados durante el procedimiento del programa en cada respectivo gráfico. Las funciones se mandan a llamar conforme sea solicitado con acciones dentro del programa. Ver Apéndice N.

n) Función utilizada para mostrar los picos detectados en la etapa de calibración o

aproximación en el gráfico “UIAxes3” dependiendo de la etapa en la que se encuentre. Ver Apéndice Ñ.

o) La siguiente función fue creada para promediar los picos máximos y mínimos por separados, con esto se logra obtener los valores necesarios para convertir la señal aproximada a DC de valores en “volts” a un valor de “píxel” o coordenada en un campo de visión con cierta resolución establecida. Ver Apéndice O.

p) El siguiente bloque de código fue diseñado para ejecutar cada una de las funciones previamente presentadas en el orden secuencial que se mostrará a continuación, antes, es importante mencionar que dichas funciones se ejecutarán de acuerdo al caso que se invoque, dichos casos son llamados cuando se realiza alguna acción en alguno de los botones de la interfaz. Lo mencionado previamente se puede entender con más claridad realizando los pasos descritos en la secciones 3.4 y 3.5. Ver Apéndice P.

q) Finalmente, se muestra el código para el manejo de eventos durante el uso de la interfaz de usuario (app). En la presente sección se destacan las funcionalidades del algoritmo de procesamiento de señales, sin embargo, no se describen las líneas de código para el desarrollo de la interfaz de usuario (app). Ver Apéndice Q.

Resultados

En la siguiente sección se detallará cómo fueron obtenidos los resultados de 3 rutinas visuales en donde un indicador (cuadrado) fue desplegado y con el tiempo fue cambiando de posición para su seguimiento visual a través de movimientos sacádicos. Cada rutina representa un tipo distinto de desplazamiento en donde varían las amplitudes y direcciones en cada una de ellas.

5.1. Descripción de las rutinas de movimiento ocular

Las rutinas desplegadas serán explicadas a continuación:

1. En la primera rutina, el indicador se posicionó en tres ubicaciones diferentes dentro de la pantalla en el eje horizontal, para tener una idea más clara, véase la Figura 5.1.

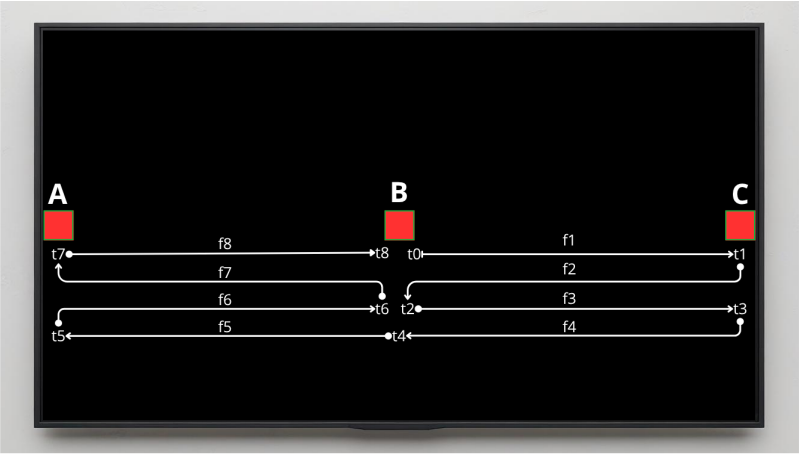


Figura 5.1: Representación gráfica de la rutina 1.

La rutina que se muestra en la Figura 5.1 en donde un símbolo cuadrado comienza en la posición B-t0, posteriormente se desplaza al punto C-t1 a través del flujo 1 (f1) manteniéndose en esa posición aproximadamente 10 segundos, posteriormente el cuadrado se regresa a B-t2 (f2) manteniéndose en esta posición el tiempo anteriormente mencionado.

Tras el segundo movimiento (f2), repitió nuevamente el patrón descrito en el párrafo anterior trazando así el movimiento f3 y f4, que es lo mismo que desplazarse del punto B-t2 al C-t3 y después regresar al B-t4. Después del cuarto movimiento (B-t4) se repite el patón descrito con anterioridad, en esta ocasión hacia el lado contrario, es decir entre B y A, de modo que el indicador parte de B-t4 a A-t5 en el flujo 5 (f5), después de eso, el indicador regresa al centro B-t6 por medio de la dirección dada por f6. Se repite el movimiento de centro a izquierda (f7) y de izquierda a centro (f8) para concluir con la rutina regresando al centro (B-t8).

El patrón anterior se puede representar en señal EOG-DC ideal, en donde los valores de la señal en la referencia corresponden a la posición del indicador en B (centro), y los valores de la señal positivos indican un desplazamiento a la derecha posicionando él indicador en C, mientras que los valores de la señal negativos representan desplazamientos a la izquierda, posicionando el indicador en A, dicha señal se muestra en la Figura 5.2.

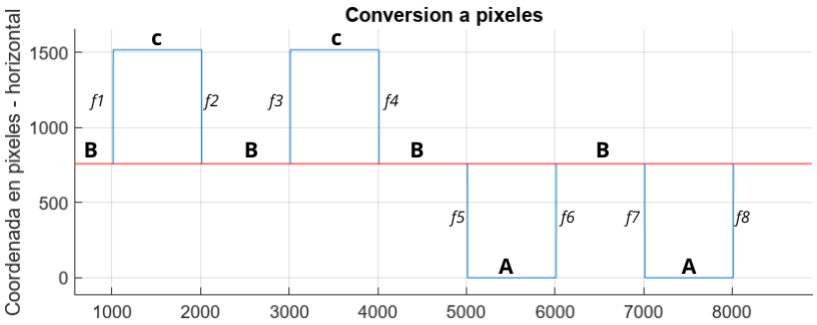


Figura 5.2: Señal representativa de la primera rutina.

En la rutina 1 no hay desplazamientos entre extremos opuestos, solamente existen desplazamientos entre un extremo al centro del campo de visión.

2. La Figura 5.3 representa la segunda rutina creada para la evaluación del algoritmo, a continuación, será explicada más a detalle.

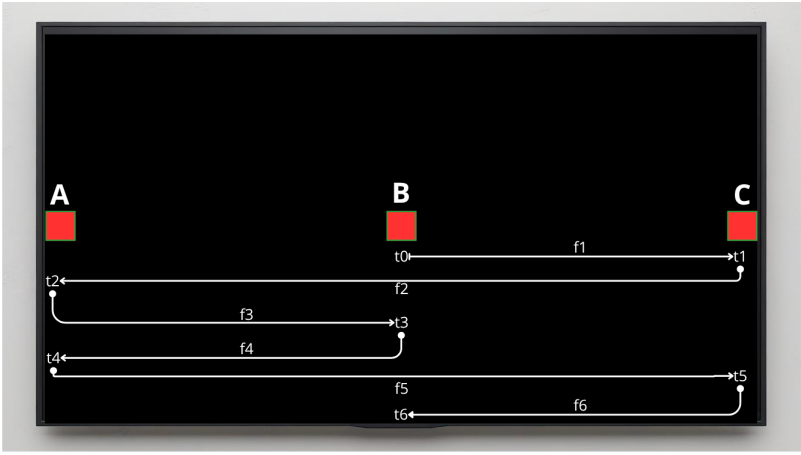


Figura 5.3: Representación gráfica de la rutina 2.

La rutina mostrada en la Figura 5.3 comienza en con el indicador en la posición B-t0 y luego se mueve al punto C-t1 a través del flujo f1, marcando así su primer cambio de posición. Después, el indicador se traslada al extremo opuesto A-t2 a través de f2 para que, después de 10 segundos aproximadamente, el indicador se mueva al punto B-t3 indicado por el flujo 3 (f3), de modo que concluye el primer ciclo completo. A partir del punto B-t3 se realiza un ciclo de desplazamientos a diferencia de que en esta ocasión se comienza desplazándose al punto A-t4 siguiendo a f4. Posteriormente, el indicador pasa por f5, llegando finalmente a C-t5. Pasado un tiempo en C-t5, el indicador se traslada a la posición central

B-t6 indicado por f6.

La señal que representa los desplazamientos explicados en la descripción de la Figura 5.3 puede verse en la Figura 5.4.

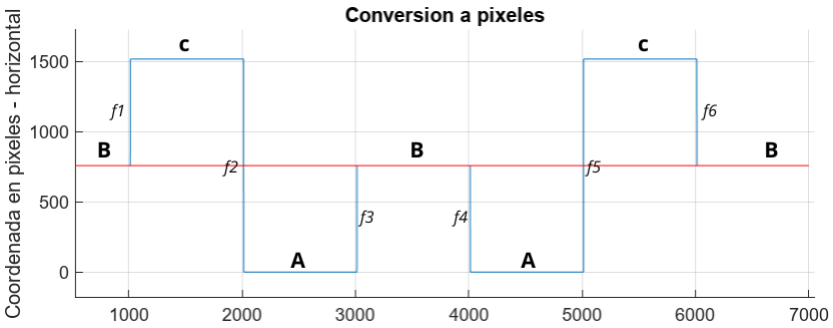


Figura 5.4: Señal representativa de la segunda rutina.

En la segunda rutina se presentan amplitudes mayores a otras, incluso sin detenerse el indicador en la posición central B, al igual que la primer rutina, las amplitudes positivas de la señal que van de un valor menor a uno mayor, indican desplazamientos hacia la derecha, por lo contrario, valores mas positivos a mas negativos, indican desplazamientos a la izquierda. La Figura 5.4 comienza con el indicador en el centro del campo de visión, con el tiempo este es desplazado hacia el extremo derecho de la pantalla y, poco tiempo después, este se traslada hacia el extremo izquierdo para finalmente regresar al centro.

3. La tercer rutina se representa en la Figura 5.5, el indicador se desplaza entre movimientos que van del centro de la pantalla hacia los extremos y movimientos que van de extremo a extremo.

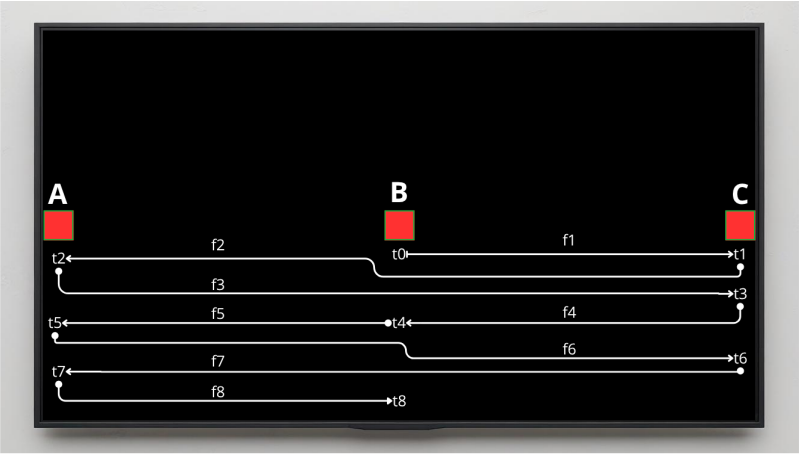


Figura 5.5: Representación gráfica de la rutina 3.

La tercer rutina corresponde a la mostrada en la Figura 5.5, y comienza con el indicador en una posición inicial en el centro de la pantalla (B-t0) y se desplaza hacia la derecha hasta el punto (C-t1) siguiendo el flujo 1 (f1). Esperado un aproximado de 10 segundos con la vista fija en la posición C-t1, el indicador se trasladó al extremo opuesto en la pantalla, llegando así al punto (A-t2) siguiendo el flujo de f2, para posteriormente, después de mantenerse fija en ese punto durante unos 10 segundos, regresar al punto C-t3 a través de f3 y esperar 10 segundos antes de regresar al centro hacia B-t4 a través de f4 completando así el primer ciclo de movimientos completos. El siguiente ciclo parte desde el centro B-t4 dirigiéndose el indicador inicialmente hacia la izquierda, al punto A-t5 siguiendo el flujo 5 (f5), pasando unos segundos el indicador es desplazada a la posición C-t6 pasando por f6 y, pasados unos segundos, el indicador se desplaza a A-t7 con la amplitud dada por f7, finalizando con el ciclo completo regresando al centro B-t8.

La forma de la señal ideal que representa la rutina anterior es la mostrada en la Figura 5.6.

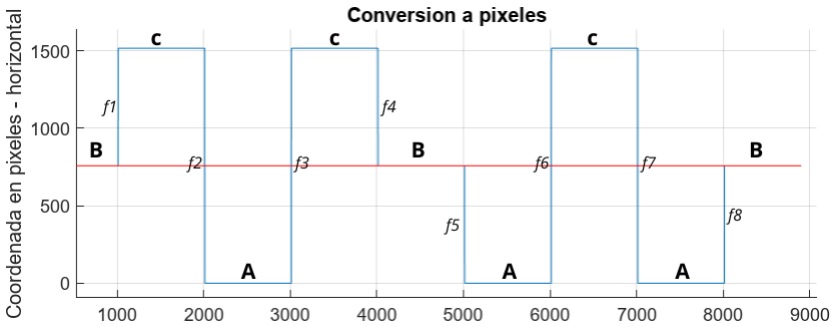


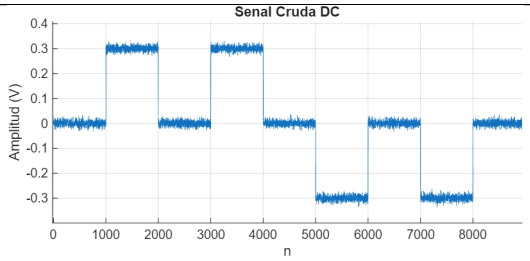
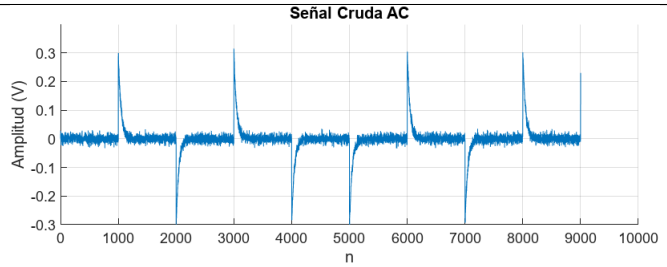
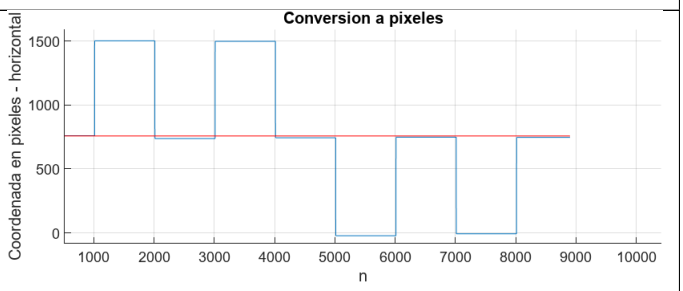
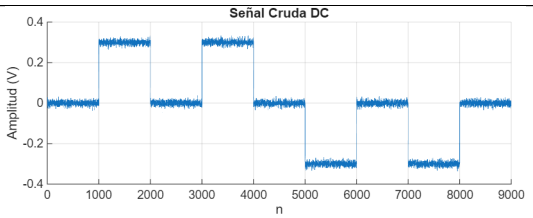
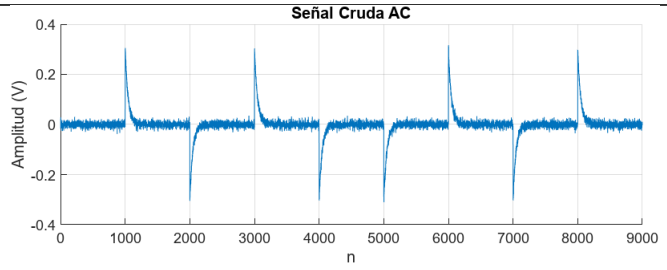
Figura 5.6: Señal representativa de la tercer rutina.

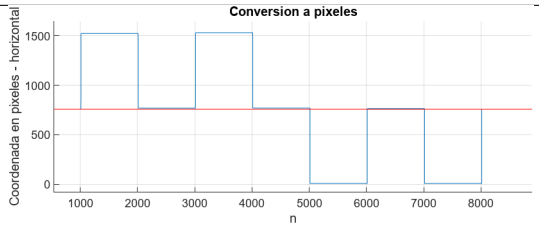
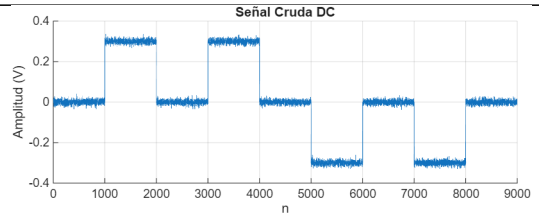
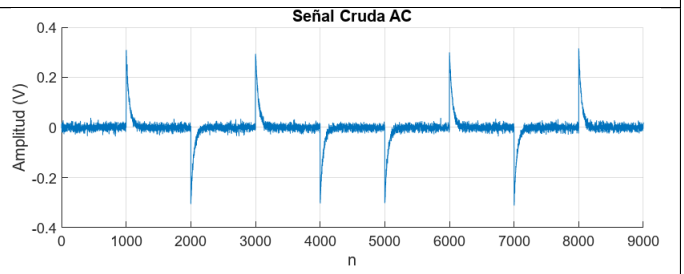
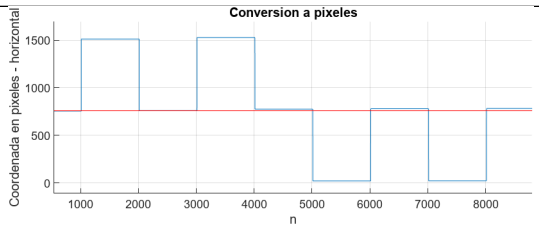
El indicador comienza desde el centro y empieza a desplazarse hacia el extremo derecho del campo de visión, después el indicador se dirige al extremo opuesto estando en el lado izquierdo para posteriormente regresar al extremo derecho nuevamente, pasados unos segundos, el indicador regresa al centro del campo de visión. El ciclo anterior se repite con la diferencia que ahora hacia las direcciones opuestas.

5.2. Resultados de la primer rutina

Para la obtención de los resultados se compararon los posicionamientos del indicador en la señal sin ruido representativa de la primera rutina (I1) contra las tres señales creadas artificialmente con ruido gaussiano (S1, S2, S3) y la señal con ruido adquirida por parte del sistema de EOG (R1). En la Tabla 5.1 se plasman las tres señales creadas artificialmente que siguen el comportamiento de la primer rutina, y en la Tabla 5.2 se muestra la señal adquirida con el sistema de EOG. Tanto las señales artificiales como la real, se representan en su forma de DC y AC. Como extra se añadieron las aproximaciones resultantes de las mismas señales (reconstrucción de AC2DC).

Tabla 5.1: Señales obtenidas siguiendo la primer rutina. Donde AC2DC corresponde a la señal reconstruida utilizando un valor de k=25.

S1	
DC	
AC	
AC2DC	
S2	
DC	
AC	

AC2DC	
S3	
DC	
AC	
AC2DC	

La tabla 5.1 presenta el resultado de las tres señales sintéticas en donde visualmente se observa una buena aproximación comparando las posicione centrales contra la referencia (línea roja).

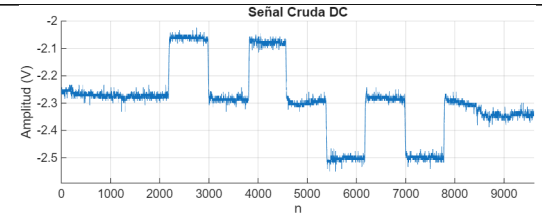
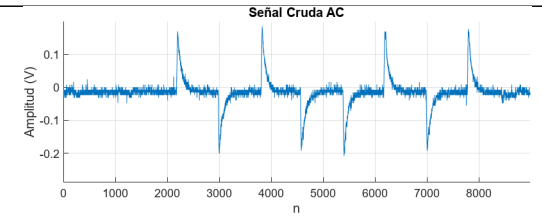
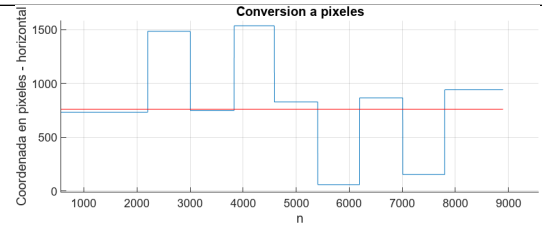
Para obtener el resultado de la aproximación a DC en píxeles de la señal real, se utilizaron los valores vistos en la Figura 5.7 y en la Figura 5.8. El parámetro “Max” indica el valor en voltaje en los promedios de los picos positivos detectados al momento de realizar un movimiento ocular del punto B al punto C y también del punto A al punto B en el campo de visión. El valor de “Min” fue obtenido realizando el cálculo del promedio de los picos negativos detectados al realizar un movimiento ocular del punto

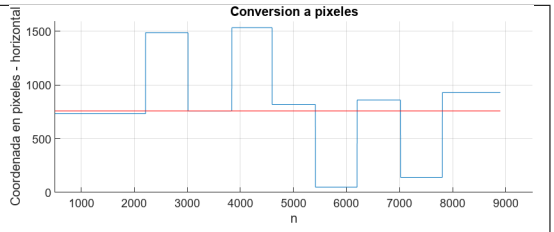
C al punto B y también los picos creados cuando se miraba del punto B al punto A. Los valores de la Figura 5.7 y la Figura 5.8 varían entre sí debido a que se probó filtrando la señal con distintos valores del parámetro "k", la primera de estas es $k = 25$ y la segunda es $k = 50$.

Figura 5.7: Parámetros obtenidos en la etapa de calibración en la primer rutina (útil para conversión a coordenadas del campo de visión). Uso para AC2DC-25k.

Figura 5.8: Parámetros obtenidos en la etapa de calibración en la primer rutina (útil para conversión a coordenadas del campo de visión). Uso para AC2DC-50k.

Tabla 5.2: Señales obtenidas siguiendo la primer rutina. Donde AC2DC-25k corresponde a la señal resconstruida utilizando un valor de k=25 y AC2DC-50k cuando el valor de k=50.

R1	
DC	
AC	
AC2DC-25k	

AC2DC-50k	

Para la Tabla 5.2 se realizó la aproximación de la señal DC sin el componente de la deriva a partir de la señal de AC, realizando el suavizado de la señal con un filtro de media móvil de ventanas rectangulares de 25 y 50 datos.

Al comparar lo obtenido visualmente en la señal real contra las sintéticas, puede observarse que la aproximación es menos exacta y precisa, y que con la ventana de 50 datos se produjeron errores menores.

Lo cual es natural debido a que las señales reales presentan imperfecciones debido al ruido ambiental, artefactos y errores de procesamiento. Sin embargo, se presume que con éxito se logró una reconstrucción DC sin componente de deriva que permite calcular la dimensión en "x" de la coordenada de la posición horizontal en la cual se está posicionando la vista del usuario.

5.2.1. Resultados del posicionamiento ocular en la primer rutina tras realizar movimientos

Las señales S1, S2, S3 y R1 fueron suavizadas y posteriormente fue llevada a cabo la aproximación de la señal DC sin el problema del componente que lleva a la deriva a la señal. Los resultados de las aproximaciones tras haber realizado la conversión de voltaje a coordenadas del campo de visión, se observan en la Tabla 5.3 los valores en pixeles de las coordenadas del posicionamiento ocular en los tiempos (t) en el campo de visión resultantes de cada desplazamiento del indicador.

Tabla 5.3: Tabla de resultados en coordenadas horizontales de un monitor 1516p horizontal

Señal \Posición	**t0**	**t1**	**t2**	**t3**	**t4**	**t5**	**t6**	**t7**	**t8**
I1 - k25	**758**	**1516**	**758**	**1516**	**758**	**0**	**758**	**0**	**758**
S1 - k25	760	1501	737	1497	744	-22	748	-7	747
S2 - k25	758	1523	769	1531	770	8	764	8	758
S3 - k25	755	1512	760	1528	775	20	779	7	761
R1 - k25	732	1484	748	1536	827	85	813	151	940
I1 - k50	**758**	**1516**	**758**	**1516**	**758**	**0**	**758**	**0**	**758**
S1 - k50	760	1507	739	1498	746	-14	754	-6	745
S2 - k50	759	1517	763	1525	763	-3	762	2	753
S3 - k50	754	1514	767	1524	770	183	787	27	787
R1 - k50	733	1487	757	1533	818	89	860	141	931

En la Tabla 5.3, lo marcado con verde (I1) indica los valores esperados para cada desplazamiento en cada instante de "t" al momento de llevar a cabo la aproximación a DC. Las señales con ruido que fueron utilizadas para comparar contra I1 fueron S1, S2, S3 y R1.

Tabla 5.4: Los valores representan el error absoluto entre la señal I1 y cada señal comparada (S1, S2, S3, R1) en cada instante de tiempo.

Comparación	**t0**	**t1**	**t2**	**t3**	**t4**	**t5**	**t6**	**t7**	**t8**
I1 vs S1 - k25	2	15	21	19	14	22	10	7	11
I1 vs S2 - k25	0	7	11	15	12	8	6	8	0
I1 vs S3 - k25	3	4	2	12	17	20	19	21	24
I1 vs R1 - k25	26	32	30	20	69	58	107	151	182
I1 vs S1 - k50	2	9	19	18	12	14	4	6	13
I1 vs S2 - k50	1	5	9	11	13	2	5	2	5
I1 vs S3 - k50	4	2	3	12	18	25	27	27	29
I1 vs R1 - k50	25	29	1	17	60	49	102	141	173

La comparación presente en la Tabla 5.4 de cada una de las señales respecto a I1 indican que la señal real acumuló un mayor error al ir incrementando los desplazamientos oculares. Las señales sintéticas con ruido no asemejan el error que se consiguió con la señal real dado que en esta última, el error fue más alto.

5.2.2. Resultados de la amplitud del desplazamiento del posicionamiento ocular en cada movimiento

Entre cada posición en la que se situaba el indicador se realizó un movimiento al cual le corresponde una amplitud específica, en el caso de la primer rutina, la amplitud de trasladar el indicador del centro hacia un extremo o viceversa es de 758 píxeles dentro del campo de visión utilizado. En la Tabla 5.5 son presentadas las amplitudes entre cada posición del indicador.

Tabla 5.5: Amplitud de desplazamiento en piíxeles en el campo de visión (el valor esperado es 758).

Señal \Movimiento	**f1 (t1–t0)**	**f2 (t2–t1)**	**f1 (t1–t0)**	**f3 (t3–t2)**	**f4 (t4–t3)**	**f5 (t5–t4)**	**f6 (t6–t5)**	**f7 (t7–t6)**
S1 - k25	741	764	760	753	766	770	755	754
S2 - k25	765	754	762	761	762	756	756	750
S3 - k25	757	752	768	753	755	759	758	761
R1 - k25	752	736	788	709	769	807	714	789
S1 - k50	747	768	759	752	760	768	760	751
S2 - k50	758	754	762	754	762	760	756	751
S3 - k50	760	753	763	754	752	765	756	760
R1 - k50	754	730	776	715	769	811	719	790

Los resultados de la diferencia en la amplitud de la señal en cada desplazamiento con respecto a la amplitud esperada de 758, puede verse en la Tabla 5.6.

Tabla 5.6: Error absoluto de la amplitud del desplazamiento respecto al valor esperado (758).

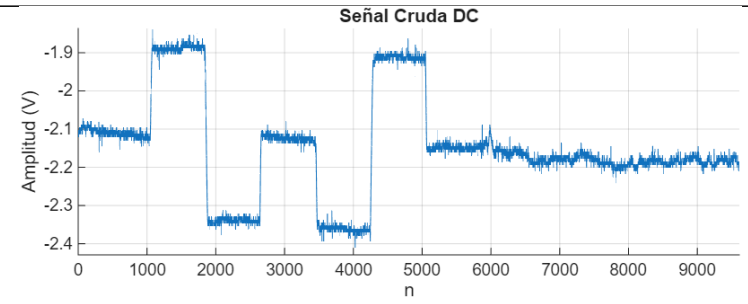
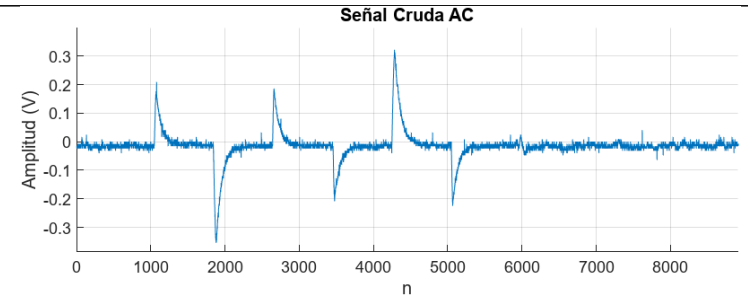
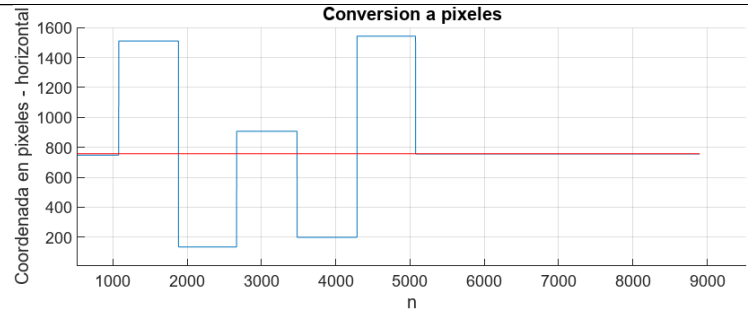
Señal \Movimiento	**f1 (t1–t0)**	**f2 (t2–t1)**	**f3 (t3–t2)**	**f4 (t4–t3)**	**f5 (t5–t4)**	**f6 (t6–t5)**	**f7 (t7–t6)**	**f8 (t8–t7)**
S1 – k25	17	6	2	5	8	12	3	4
S2 – k25	7	4	3	4	3	2	2	8
S3 – k25	1	6	10	5	3	1	0	2
R1 – k25	6	22	30	49	11	49	44	31
S1 – k50	11	10	1	6	2	2	7	3
S2 – k50	2	4	5	4	4	4	3	7
S3 – k50	2	5	4	3	7	7	2	2
R1 – k50	4	28	18	43	11	53	39	32

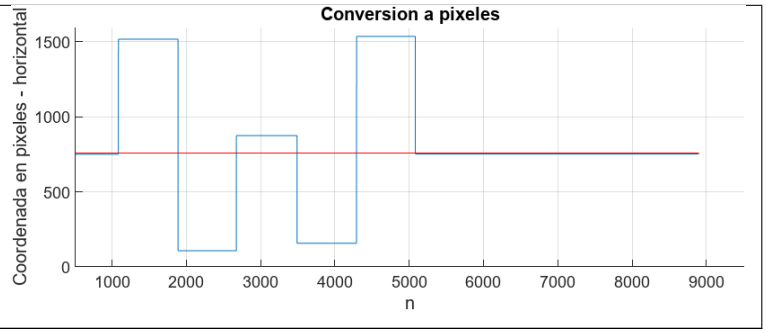
El error en cada desplazamiento es menor en las señales sintéticas contra el valor esperado al momento de realizar la aproximación, en cambio, en la señal real, el error es más alto en algunos movimientos oculares, pero no en todos.

5.3. Resultados de la segunda rutina

Para la obtención de resultados de la segunda rutina de movimiento ocular, se compararon los posicionamientos del indicador en la señal sin ruido (como fue visto en la Sección 5.2) pero representativa a la segunda rutina (I2) contra la señal adquirida con el sistema de EOG (R2). En la Tabla 5.7 se plasma la señal adquirida en su forma DC y AC, además de mostrarse el resultado de la aproximación a DC en valor de píxeles correspondientes al campo de visión. La aproximación fue realizada en señales que fueron suavizadas con el filtro de media móvil con ventana de 25 y 50 datos (AC2DC-25k y AC2DC-50k).

Tabla 5.7: Señales obtenidas siguiendo la segunda rutina del movimiento ocular. Donde AC2DC corresponde a la señal reconstruida.

	R2
DC	
AC	
AC2DC-25k	

AC2DC-50k	

La Tabla 5.7 contiene la visualización de la señal correspondiente a los movimientos oculares en su forma de DC y AC tal cual fue adquirida del sistema de adquisición siguiendo la segunda rutina. De igual forma se muestra el gráfico que corresponde a la señal después del procesado de la señal AC para aproximarla a un valor parecido al de DC. Los resultados visuales de la aproximación fueron aplicando suavizado de media móvil en la señal AC con una ventana rectangular de 25 datos (25k) y 50 datos (50k).

El resultado se encuentra en coordenadas dentro del campo de visión, estos valores fueron obtenidos con la normalización de los datos "Max" y "Min" vistos en la Figura 5.9 utilizados cuando se aplica el filtro con ventana $k = 25$ y en la Figura 5.10 cuando $k = 50$ (procedimiento similar que en la Sección 5.2).

Figura 5.9: Parámetros obtenidos en la etapa de calibración en la segunda rutina (útil para conversión a coordenadas del campo de visión). Uso para AC2DC-25k.

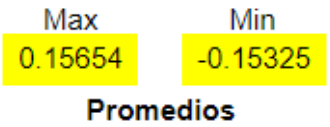


Figura 5.10: Parámetros obtenidos en la etapa de calibración en la segunda rutina (útil para conversión a coordenadas del campo de visión). Uso para AC2DC-50k.

5.3.1. Resultados del posicionamiento ocular en la segunda rutina tras realizar movimientos sacádicos

Los siguientes son los resultados de la segunda rutina (I2) en cada momento "t", en esta sección se compararon los valores esperados contra lo obtenido en la aproximación de la segunda rutina (véase la Tabla 5.8).

La señal R2 fue utilizada en dos experimentos, en uno de ellos fue filtrada con un valor de $k = 25$ y posteriormente se realizó la aproximación (R2 - k25). En el segundo experimento se aplicó un valor de $k = 50$ (R2 - k50), esto con el objetivo de comparar resultados en las aproximaciones a DC realizadas y observar que tanto se aproximan a lo esperado filtrando con diferente tamaño de ventana.

Tabla 5.8: Tabla de resultados de los movimientos oculares en para la segunda rutina en coordenadas horizontales de un monitor 1516p horizontal

Señal \Movimiento	**t0**	**t1**	**t2**	**t3**	**t4**	**t5**	**t6**
I2	**758**	**1516**	**0**	**758**	**0**	**1516**	**758**
R2 - k25	748	1511	135	907	199	1543	752
R2 - k50	750	1516	108	874	157	1535	752

En la Tabla 5.8 fueron vaciados los valores correspondientes a la coordenada en el eje "x" del campo de visión, los cuales indican el posicionamiento ocular en cada momento de "t" de I2 y R2.

Tabla 5.9: Error absoluto en cada "t" para los movimientos de la segunda rutina.

Señal \Movimiento	**t0**	**t1**	**t2**	**t3**	**t4**	**t5**	**t6**
I2 vs R2 - k25	10	5	135	149	199	27	2
I2 vs R2 - k50	8	0	108	116	157	19	6

Los resultados obtenidos en la Tabla 5.9 indican que el valor esperado es muy alto a partir de "t2" en ambos casos, cuando la señal en $k = 25$ y $k = 50$. De igual forma se logra apreciar que el error aumenta hasta "t4", posterior a eso, el error acumulado disminuye.

5.3.2. Resultados de la amplitud del desplazamiento del posicionamiento ocular en cada movimiento

En la presente sección se plasman los valores de píxeles esperados en cada momento en el que se realizó un movimiento sacádico debido al seguimiento de la segunda rutina.

Entre cada posición en la que se situaba el indicador se realizó un movimiento al cual le corresponde una amplitud específica, en el caso de la segunda rutina, la amplitud de trasladar el indicador del centro hacia un extremo o viceversa es de 758 píxeles dentro del campo de visión utilizado, mientras que desplazar el indicador de un extremo a otro la amplitud correspondiente es de 1516 píxeles. En la Tabla 5.10 son presentadas las amplitudes entre cada posición del indicador.

Tabla 5.10: Amplitud de desplazamiento en valor de coordenada del campo de visión.

Señal \Movimiento	**f1 (t1–t0)**	**f2 (t2–t1)**	**f3 (t3–t2)**	**f4 (t4–t3)**	**f5 (t5–t4)**	**f6 (t6–t5)**
R2 - k25	763	1376	772	708	1344	787
R2 - k50	766	1408	766	717	1378	783

En la Tabla 5.10 fueron vaciados los resultados de los movimientos sacádicos al momento de haber sido realizados en la segunda rutina, estos son representados como la diferencia entre la posición actual y la anterior, por ejemplo “t1” y “t0” (f1). El valor esperado en f1, f3, f4 y f6 es de 758 mientras que en f2 y f5 es de 1516.

Tabla 5.11: Error absoluto en cada movimiento sacádico realizado respecto al valor esperado.

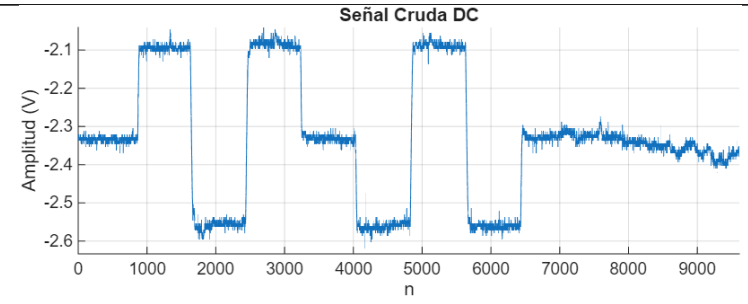
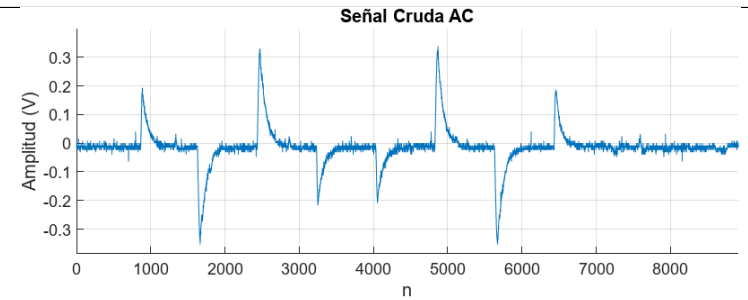
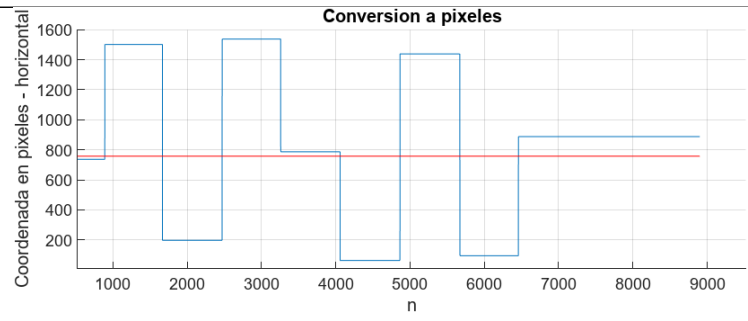
Señal \Movimiento	**f1 (t1–t0)**	**f2 (t2–t1)**	**f3 (t3–t2)**	**f4 (t4–t3)**	**f5 (t5–t4)**	**f6 (t6–t5)**
R2 - k25	5	140	14	50	172	29
R2 - k50	8	108	8	41	138	25

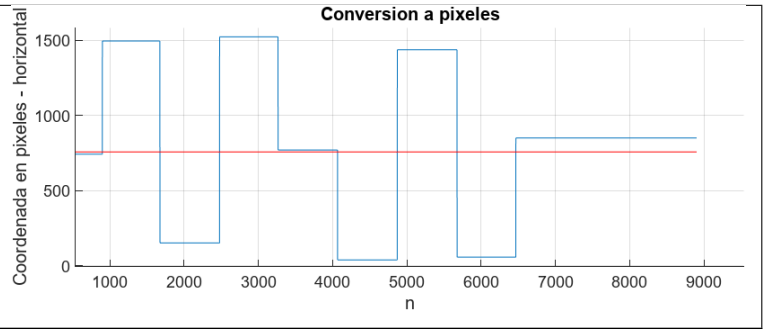
La Tabla 5.11 indica una serie de movimientos sacádicos en donde se espera el error mínimo posible, se aprecia que el error en la segunda rutina pasa de las centenas en f2 y f5, es decir, cuando se realizan movimientos sacádicos partiendo de un extremo del campo de visión hacia el otro extremo.

5.4. Resultados de la tercer rutina

La sección actual detalla los resultados correspondientes a la tercer rutina de movimientos oculares (I3). La Tabla 5.12 contiene gráficas de las señales obtenidas a partir del sistema de adquisición de señales correspondientes a la tercer rutina de movimiento ocular (R3) en su forma de DC y AC y además, se incluye el resultado final de la aproximación de la señal en DC a partir de la señal AC utilizando el filtro de media móvil con una ventana de $k = 25$ y también con $k = 50$.

Tabla 5.12: Señales obtenidas siguiendo la tercer rutina del movimiento ocular. Donde AC2DC corresponde a la señal resconstruida.

R3	
DC	
AC	
AC2DC-25k	

AC2DC-50k	

Visualmente los resultados presentes en la Tabla 5.12 (AC2DC-25k y AC2DC-50k) comienzan un poco desviados de la referencia en ambos casos, cuando la señal de AC se suavizó con el filtro de media móvil con una ventana de 25 y 50 datos. De igual forma, se observa que el error incrementa en cada movimiento sacádico, llegando así hasta el último instante, donde se aprecia que el valor de la línea azul se encuentra por encima de la roja con mucha diferencia.

Para realizar la conversión de voltaje a píxeles correspondientes al monitor y con ello la aproximación se utilizaron los valores que se muestran en la Figura 5.11 para AC2DC de la tercer rutina utilizando el filtro con 25 datos y lo mostrado en la Figura 5.12 al haber aplicado el filtro con una ventana de 50 datos.

Figura 5.11: Parámetros obtenidos en la etapa de calibración en la tercer rutina (útil para conversión a coordenadas del campo de visión). Uso para AC2DC-25k.

Figura 5.12: Parámetros obtenidos en la etapa de calibración en la segunda tercer (útil para conversión a coordenadas del campo de visión). Uso para AC2DC-50k.

5.4.1. Resultados del posicionamiento ocular en la tercer rutina tras realizar movimientos

En la presente subsección se presentan los resultados de los movimientos oculares respectivos a la tercer rutina (I3) en cada momento de "t". Fueron comparados los resultados obtenidos de la señal adquirida con el sistema de EOG habiendo realizado el seguimiento del indicador en la tercer rutina (R3) contra los valores esperados en cada "t" (véase la Tabla 5.13).

La señal R3 fue utilizada en dos experimentos, en uno de ellos fue filtrada con un valor de $k = 25$ y posteriormente se realizó la aproximación (R2 - k25). En el segundo experimento se aplicó un valor de $k = 50$ (R2 - k50), esto con el objetivo de comparar que tan exacto fue cada posicionamiento con los dos tipos de configuraciónes en el filtro.

Tabla 5.13: Tabla de resultados en coordenadas horizontales de un monitor 1516p horizontal.

Señal\Movimiento	**t0**	**t1**	**t2**	**t3**	**t4**	**t5**	**t6**	**t7**	**t8**
I3	**758**	**1516**	**0**	**1516**	**758**	**0**	**1516**	**0**	**758**
R3 - k25	737	1502	198	1538	787	64	1439	95	888
R3 - k50	747	1497	152	1524	771	39	1438	58	851

En la Tabla 5.13 son mostrados los valores correspondientes a la coordenada en el eje "x" del posicionamiento ocular en cada momento de "t" en R3. El valor de I3 es lo que se espera para R3.

Tabla 5.14: Error absoluto en cada "t" para los movimientos de la tercera rutina.

Comparación	**t0**	**t1**	**t2**	**t3**	**t4**	**t5**	**t6**	**t7**	**t8**
I3 vs R3 - k25	21	14	198	22	29	64	77	95	130
I3 vs R3 - k50	11	19	152	8	13	39	78	58	93

La relación de lo esperado contra lo conseguido (véase la Tabla 5.14) indica un error muy alto al mantener la vista en un punto de cada momento en "t" para la tercer rutina. El mayor error se encontró en "t2" en ambos casos, cuando se filtró con una ventana de 25 datos como cuando se realizó lo mismo pero con una ventana de 50 datos en R3. Aparentemente, en los movimientos sacádicos mayores el error fue alto.

5.4.2. Resultados de la amplitud del posicionamiento ocular en cada movimiento

Se presentan los resultados correspondientes a la amplitud en cada movimiento sacádico (f) habiendo realizado el seguimiento de la tercer rutina de movimiento ocular. El resultado es expresado con píxeles de un campo de visión definido.

Las amplitudes de cada movimiento sacádico variaban conforme el desplazamiento del indicador en la pantalla. En el caso de la tercer rutina, existieron amplitudes de 758 píxeles al tratarse de movimientos del centro hacia un extremo de la pantalla y por otro lado, se presentaron amplitudes correspondientes a los 1516 píxeles cuando se realizaban movimientos de extremo a extremo (parecido a lo visto en la segunda rutina).

Tabla 5.15: Amplitud de desplazamiento en valor de coordenada del campo de visión (el valor esperado es 758 para f1, f4, f5 y f8 y 1516 para f2, f3, f6 y f7).

Señal \Movimiento	**f1 (t1–t0)**	**f2 (t2–t1)**	**f3 (t3–t2)**	**f4 (t4–t3)**	**f5 (t5–t4)**	**f6 (t6–t5)**	**f7 (t7–t6)**	**f8 (t8–t7)**
R3 - k25	765	1304	1340	751	723	1375	1344	793
R3 - k50	750	1345	1372	753	732	1399	1380	793

En la Tabla 5.15 se muestran las amplitudes entre cada posición del indicador. Fueron depositados los resultados de los movimientos sacádicos al momento de haber realizado la aproximación a DC con la tercer rutina de movimiento ocular. Los resultados esperados para f1, f4, f5 y f8 son de 758 píxeles mientras que para f2, f3, f6 y f7 son de 1516 píxeles.

Tabla 5.16: Error absoluto en cada fase del movimiento.

Señal \Movimiento	**f1 (t1–t0)**	**f2 (t2–t1)**	**f3 (t3–t2)**	**f4 (t4–t3)**	**f5 (t5–t4)**	**f6 (t6–t5)**	**f7 (t7–t6)**	**f8 (t8–t7)**
R3 - k25	7	212	176	7	35	141	172	35
R3 - k50	13	171	144	5	26	117	136	35

La Tabla 5.16 presenta el error obtenido de cada uno de los movimientos sacádicos realizados en la tercer rutina contra lo esperado. Los errores más altos fueron presentes en los desplazamientos con mayor amplitud (sacádicos correspondientes a movimientos de extremo a extremo del monitor), de modo que pasan las centenas de píxeles, estos son presentes en f2, f3, f6 y f7, en donde los valores esperados son de 1516 píxeles, es

decir, cuando se realizan movimientos sacádicos partiendo de un extremo del campo de visión hacia el otro extremo.

5.5. Análisis del error absoluto versus resolución

Se procesaron las señales provenientes de observar tres patrones, obteniéndose en total 22 mediciones de posicionamiento de la vista después de realizar los movimientos oculares sacádicos horizontales. Se observó que el procesamiento de las señales con el filtro de media móvil con $k = 50$, presentó mejores resultados. Comparándose las coordenadas de los patrones contra las mediciones se obtuvo el mínimo error absoluto de cero y máximo de 173, los cuales pueden ser atribuidos a la capacidad del usuario de enfocar el indicador que lo guío en el patrón del movimiento ocular, al sistema de adquisición y procesamiento de señales, así como al método de reconstrucción, el cual permite que se vaya acumulando y compensando la trazabilidad de las mediciones. Bajo el escenario de la experimentación con un monitor de 1516 píxeles y en búsqueda de la excelencia para una aplicación de asistencia, se proyecta una incertidumbre de ± 173 píxeles, ofertando 3 opciones de asistencia como se ilustra en la Figura 5.13.

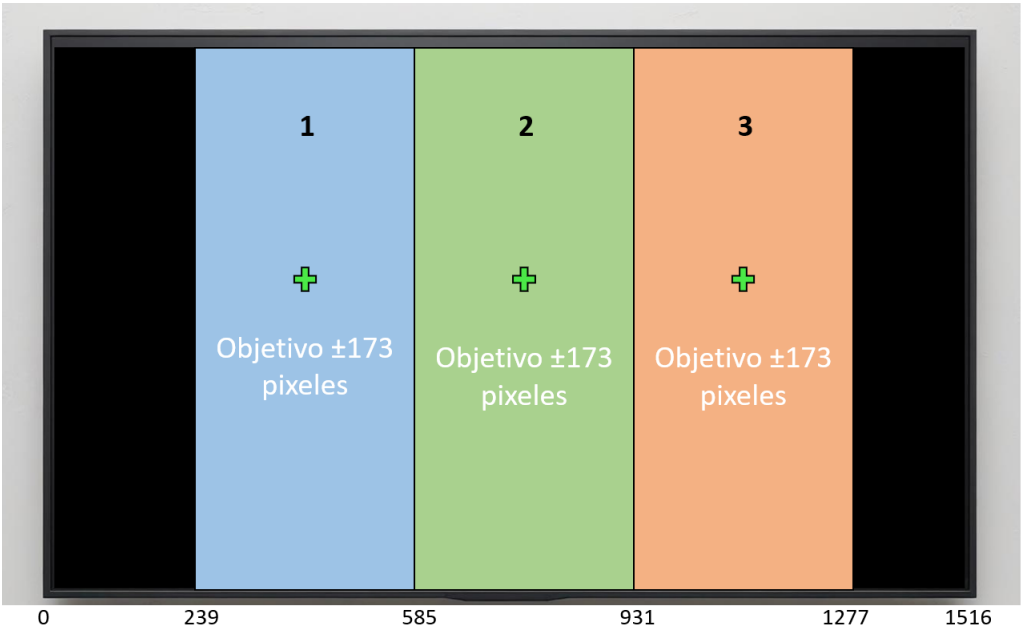


Figura 5.13: Descripción gráfica de interfaz con 3 objetivos con ± 173 píxeles de rango para asegurar la selección.

El promedio del error absoluto de las mediciones arrojo valor de 68 píxeles, con una desviación estándar de 59 píxeles, bajo este escenario sería posible ofertar 11 opciones de asistencia como se ilustra en la Figura 5.14. Los resultados del promedio y la desviación estándar se deben a que la distribución de los datos es muy dispersa.

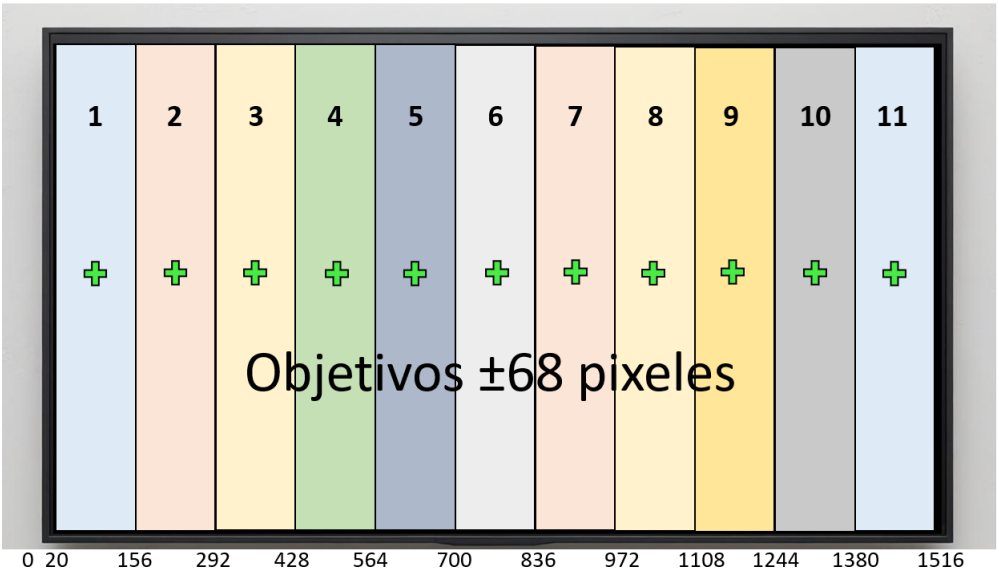


Figura 5.14: Descripción gráfica de interfaz con 11 objetivos con ± 68 píxeles de rango para la selección, sin asegurar se logre el objetivo.

Es decir, el 50 % de las mediciones presentan un error absoluto de medición menor de 35 píxeles, mientras el resto varía desde 35 hasta 173 píxeles, proveyendo un escenario de 22 opciones de asistencia como se ilustra en la Figura 5.15.

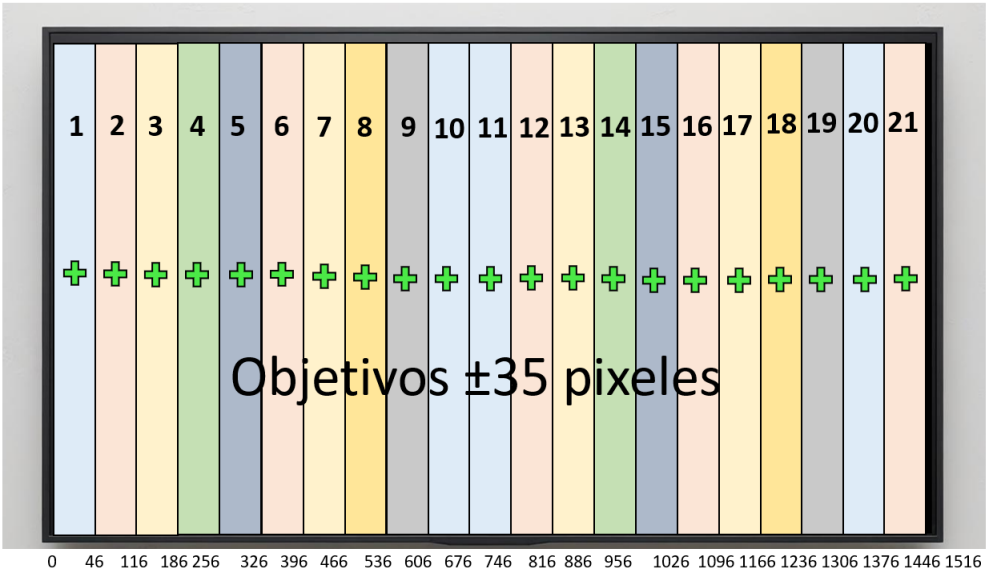


Figura 5.15: Descripción gráfica de interfaz con 21 objetivos con ± 35 píxeles de rango para la selección, sin asegurar se logre el objetivo.

Conclusiones

El trabajo de investigación desarrollado aborda de manera integral la problemática de interpretar señales EOG (electrooculografía) para su uso en el desarrollo de interfaces hombre-máquina (HMI), con énfasis en su aplicación para personas con discapacidad motriz. El proyecto se centra en los movimientos oculares sacádicos horizontales y propone una solución que supera varios de los desafíos actuales en el área, como la falta de estandarización en la interpretación de señales EOG y las limitaciones de resolución en interfaces visuales basadas en VOG (Videooculografía) bajo distintos escenarios de iluminación y posición.

Uno de los principales aportes de esta tesis es el diseño y validación de una interfaz capaz de estimar, en un aproximado a tiempo real, la posición de la mirada en coordenadas de píxeles, a partir de la reconstrucción de señales EOG transformadas desde su forma en corriente alterna (AC) hacia una representación equivalente en corriente directa (DC). Esta aproximación suprime el problema de la deriva de la señal EOG DC, lo cual es una barrera importante en el uso efectivo de EOG en aplicaciones prácticas.

La metodología incluyó la simulación inicial con señales artificiales, seguida por pruebas con datos reales obtenidos de un sistema de adquisición. La estrategia de calibración implementada demostró ser efectiva para ajustar el sistema a la fisiología individual de los usuarios, resolviendo uno de los problemas más persistentes en la interpretación de EOG: la variabilidad interindividual. Así, el sistema es adaptable y puede operar sin requerir ajustes manuales extensos, lo cual facilita su implementación práctica.

El sistema propuesto logra detectar desplazamientos con precisión suficiente para mapear la mirada en una pantalla con una resolución horizontal de hasta 1516 píxeles.

Además, la restricción de ±173 píxeles de variación permite dividir este espacio en al menos 3 cuadros distinguibles, lo que posibilita interacciones visuales simples pero funcionales, como seleccionar íconos, mover cursores o activar comandos básicos.

En términos de impacto, el desarrollo de esta tecnología representa una alternativa más económica, menos invasiva con respecto a metodos como el de bobina de busqueda [60] y con menor latencia que otros métodos basados en cámaras infrarrojas o gafas especializadas. Su bajo costo y portabilidad aumentan su aplicabilidad en contextos clínicos, educativos y domésticos. También abre nuevas oportunidades para el diseño de sistemas de control accesibles en aplicaciones de realidad aumentada, juegos, rehabilitación y asistencia personal.

En conclusión, la tesis no solo propone una solución técnica a un problema vigente, sino que también plantea una línea clara hacia futuras investigaciones, como la extensión del sistema para interpretar movimientos verticales y diagonales, así como la integración de algoritmos de inteligencia artificial para la clasificación avanzada de patrones oculares. El enfoque metodológico, la solidez técnica y la pertinencia social del trabajo lo convierten en una contribución al campo de las interfaces biomédicas y la accesibilidad tecnológica.

Bibliografía

[1] Moises Diaz Cabrera, Miguel Ángel Ferrer Ballester, María Cristina Carmona Duarte, and Réjean Plamondon. Improving handwritten signatures fluency via the lognormality principle. 2020.

[2] Fuming Fang and Takahiro Shinozaki. Electrooculography-based continuous eye-writing recognition system for efficient assistive communication systems. *PloS one*, 13(2):e0192684, 2018.

[3] Seokjue Jeong, Sunghan Lee, Jeonghwan Koh, Hyungchan An, and In cheol Jeong. Eye image and eog signal conversion via autoencoders for human-machine interaction. In *2025 IEEE International Conference on Consumer Electronics (ICCE)*, pages 1–4. IEEE, 2025.

[4] David Escobar-Valencia, Fernando Jesús Regino-Ubarnes, and Diana Yamileth Velásquez-Maldonado. Señales eog: Una revisión sobre procesamiento y aplicaciones. 2022.

[5] Palpolage Don Shehan Hiroshan Gunawardane, Raymond Robert MacNeil, Leo Zhao, James Theodore Enns, Clarence Wilfred de Silva, and Mu Chiao. A fusion algorithm based on a constant velocity model for improving the measurement of saccade parameters with electrooculography. *Sensors*, 24(2), 2024.

[6] Zheng Zeng, Linkai Tao, Jun Hu, Ruizhi Su, Long Meng, Chen Chen, and Wei Chen. Multi-scale inception-based deep fusion network for electrooculogram-based eye movements classification. *Biomedical Signal Processing and Control*, 103:107377, 2025.

[7] Dan Witzner Hansen and Qiang Ji. In the eye of the beholder: A survey of models for eyes and gaze. *IEEE transactions on pattern analysis and machine intelligence*, 32(3):478–500, 2009.

[8] Mandalapu Sarada Devi and Preeti R Bajaj. Driver fatigue detection based on eye tracking. In *2008 First International Conference on Emerging Trends in Engineering and Technology*, pages 649–652. IEEE, 2008.

[9] Iván García, Sebastián Bronte, Luis Miguel Bergasa, Noelia Hernández, Beatriz Delgado, and Matías Sevillano. Vision-based drowsiness detector for a realistic driving simulator. In *13th International IEEE Conference on Intelligent Transportation Systems*, pages 887–894. IEEE, 2010.

[10] Xia Liu, Fengliang Xu, and Kikuo Fujimura. Real-time eye detection and tracking for driver observation under various light conditions. In *Intelligent Vehicle Symposium, 2002. IEEE*, volume 2, pages 344–351. IEEE, 2002.

[11] Anil K Jain, Ruud Bolle, and Sharath Pankanti. *Biometrics: personal identification in networked society*, volume 479. Springer Science & Business Media, 2006.

[12] Christel-loic Tisse, Lionel Martin, Lionel Torres, Michel Robert, et al. Person identification technique using human iris recognition. In *Proc. Vision Interface*, volume 294, pages 294–299, 2002.

[13] Zakir Hossain, Md Maruf Hossain Shuvo, and Prionjit Sarker. Hardware and software implementation of real time electrooculogram (eog) acquisition system to control computer cursor with eyeball movement. In *2017 4th international conference on advances in electrical engineering (ICAEE)*, pages 132–137. IEEE, 2017.

[14] RG Bozomitu. Asistsys, integrated system of assistance for patients with severe neuromotor affections, 2011.

[15] RG Bozomitu. Siact, integrated system for assistance in communicating with and telemonitoring severe neuromotor disabled people, 2017.

[16] Radu Gabriel Bozomitu, Lucian Niţă, Vlad Cehan, Ioana Dana Alexa, Adina Carmen Ilie, Alexandru Păsăricǎ, and Cristian Rotariu. A new integrated system for assistance in communicating with and telemonitoring severely disabled patients. *Sensors*, 19(9):2026, 2019.

[17] Sebastian Pannasch, Jens R Helmert, Susann Malischke, Alexander Storch, and Boris M Velichkovsky. Eye typing in application: A comparison of two systems with als patients. *Journal of Eye Movement Research*, 2(4), 2008.

[18] Amer Al-Rahayfeh and Miad Faezipour. Eye tracking and head movement detection: A state-of-art survey. *IEEE journal of translational engineering in health and medicine*, 1:2100212–2100212, 2013.

[19] Andrew T Duchowski and Andrew T Duchowski. *Eye tracking methodology: Theory and practice*. Springer, 2017.

[20] Päivi Majaranta and Andreas Bulling. Eye tracking and eye-based human–computer interaction. In *Advances in physiological computing*, pages 39–65. Springer, 2014.

[21] Jonathon B Hiley, Andrew H Redekopp, and Reza Fazel-Rezai. A low cost human computer interface based on eye tracking. In *2006 International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 3226–3229. IEEE, 2006.

[22] Hari Singh and Jaswinder Singh. Human eye tracking and related issues: A review. *International Journal of Scientific and Research Publications*, 2(9):1–9, 2012.

[23] Kohei Arai and Ronny Mardiyanto. Eye-based hci with full specification of mouse and keyboard using pupil knowledge in the gaze estimation. In *2011 Eighth International Conference on Information Technology: New Generations*, pages 423–428. IEEE, 2011.

[24] Nguyen Huu Cuong and Huynh Thai Hoang. Eye-gaze detection with a single webcam based on geometry features extraction. In *2010 11th International Conference on Control Automation Robotics & Vision*, pages 2507–2512. IEEE, 2010.

[25] S Bronte, LM Bergasa, B Delgado, M Sevillano, I Garcia, and N Hernandez. Vision-based drowsiness detector for a realistic driving simulator. In *13th International IEEE Conference on Intelligent Transportation Systems (ITSC 2010) Institute of Electrical and Electronics Engineers (IEEE)*, 2010.

[26] S Saravanakumar and N Selvaraju. Eye tracking and blink detection for human computer interface. *International Journal of Computer Applications*, 2(2):7–9, 2010.

[27] Ali Bulent Usakli and Serkan Gurkan. Design of a novel efficient human–computer interface: An electrooculagram based virtual keyboard. *IEEE transactions on instrumentation and measurement*, 59(8):2099–2108, 2009.

[28] Jung-Jin Yang, Gyeong Woo Gang, and Tae Seon Kim. Development of eog-based human computer interface (hci) system using piecewise linear approximation (pla) and support vector regression (svr). *Electronics*, 7(3):38, 2018.

[29] Won-Du Chang. Electrooculograms for human–computer interaction: A review. *Sensors*, 19(12):2690, 2019.

[30] Susanna Nilsson, Torbjörn Gustafsson, and Per Carleberg. Hands free interaction with virtual information in a real environment. *Proceedings of COGAIN*, pages 53–57, 2007.

[31] Susanna Nilsson, Torbjörn Gustafsson, and Per Carleberg. Hands free interaction with virtual information in a real environment: Eye gaze as an interaction tool in an augmented reality system. *PsychNology Journal*, 7(2), 2009.

[32] Javier San Agustin, Julio C Mateo, John Paulin Hansen, and Arantxa Villanueva. Evaluation of the potential of gaze input for game interaction. *PsychNology Journal*, 7(2), 2009.

[33] Michael Dorr, Martin Böhme, Thomas Martinetz, and Erhardt Barth. Gaze beats mouse: a case study. *Proceedings of COGAIN*, pages 16–19, 2007.

[34] Michael Dorr, Laura Pomarjanschi, and Erhardt Barth. Gaze beats mouse: A case study on a gaze-controlled breakout. *PsychNology Journal*, 7(2), 2009.

[35] Oliver J Muensterer, Martin Lacher, Christoph Zoeller, Matthew Bronstein, and Joachim Kübler. Google glass in pediatric surgery: an exploratory study. *International journal of surgery*, 12(4):281–289, 2014.

[36] David Beukelman, Susan Fager, and Amy Nordness. Communication support for people with als. *Neurology research international*, 2011(1):714693, 2011.

[37] Mobina Zibandehpoor, Fatemeh Alizadehziri, Arash Abbasi Larki, Sobhan Teymouri, and Mehdi Delrobaei. Electrooculography dataset for objective spatial navigation assessment in healthy participants. *Scientific Data*, 12(1):553, 2025.

[38] Sameer Nooh, Mahmoud Ragab, Rania Aboalela, Abdullah AL AL-Ghamdi, Omar A Abdulkader, and Ghadah Alghamdi. An exploratory analysis of longitudinal artificial intelligence for cognitive fatigue detection using neurophysiological based biosignal data. *Scientific Reports*, 15(1):1–17, 2025.

[39] Nataliya Kosmyna, Karim El Adl, and Minsol Kim. Wearable pair of eeg, eog and fnirs glasses for cognitive workload detection. In *2024 IEEE 20th International Conference on Body Sensor Networks (BSN)*, pages 1–4. IEEE, 2024.

[40] Minzhong Yu, Emile R Vieta-Ferrer, Anas Bakdalieh, and Travis Tsai. The role of visual electrophysiology in systemic hereditary syndromes. *International Journal of Molecular Sciences*, 26(3):957, 2025.

[41] S Dietz-Terjung, M Jakobs, C Labeit, C Schöbel, and S Buschjäger. Beyond sleep staging: Advancing end-to-end event scoring in sleep medicine. *Pneumologie*, 79(S 01):P–118, 2025.

[42] Omer Abd AL-Sattar Mohammed AL et al. Hybrid deep learning model based on transformer encoder for sleep stages classification. *Bilad Alrafidain Journal for Engineering Science and Technology*, 4(1):113–126, 2025.

[43] Manikrao Dhore and Ankit Gaikwad. Eog-based authentication system for specially abled users. In *Fifth Congress on Intelligent Systems: CIS 2024, Volume 2*, volume 1276, page 1. Springer Nature, 2025.

[44] Suyash Sonawane, Prem Lunawat, Vedant Mungase, Siddhramashwar Pawar, Om Shinde, Komal Bidkar, and Nikhil Raj Gupta. Drowsy or sleep detector glasses for vehicle drivers. *Multidisciplinary Research*, page 29, 2025.

[45] Manikrao Dhore and Gaurav Gaikwad. An electrooculogram-based control system utilizing multidirectional eye movements for als patients. In *Congress on Intelligent Systems*, pages 321–333. Springer, 2025.

[46] Xingge Yu, Zebang Luo, Xilin Ouyang, Wenqiang Wang, Yuxuan Rao, Yulong Yuan, Zhenpeng Cai, Youfan Hu, and Li Xiang. Highly stable polymeric electrooculography electrodes for contactless human-machine interactions. *ACS sensors*, 10(4):3013–3022, 2025.

[47] Wenjing Xiong, Lin Ma, and Haifeng Li. A general dual-pathway network for eeg denoising. *Frontiers in Neuroscience*, 17:1258024, 2024.

[48] Paul Riordan-Eva. *Vaughan y Asbury: oftalmología general (18a.* McGraw Hill Mexico, 2012.

[49] GoB Arden and JH Kelsey. Changes produced by light in the standing potential of the human eye. *The Journal of physiology*, 161(2):189–204, 1962.

[50] Geoffrey B Arden and Paul A Constable. The electro-oculogram. *Progress in retinal and eye research*, 25(2):207–248, 2006.

[51] Hiroki Tamura, Masaki Miyashita, Koichi Tanno, and Yasushi Fuse. Mouse cursor control system using electrooculogram signals. In *2010 World Automation Congress*, pages 1–6. IEEE, 2010.

[52] Qiuping Ding, Kaiyu Tong, and Guang Li. Development of an eog (electro-oculography) based human-computer interface. In *2005 IEEE Engineering in Medicine and Biology 27th Annual Conference*, pages 6829–6831. IEEE, 2006.

[53] Carmen Fernanda Amaya Moisa, Carlos Eduardo Azucena Peña, Katheryn Alejandra López Salazar, and David Alejandro Vega Tario. Primer avance del proyecto de cátedra:"detector de movimiento ocular con eog".

[54] Yonatan Mari Reyes. Adquisición de características de señales mioeléctricas para uso en protésica. 2012.

[55] Md Moin Uddin Atique, Sakhawat Hossen Rakib, and Khondkar Siddique-e Rabbani. An electro-oculogram based control system. In *2016 5th International Conference on Informatics, Electronics and Vision (ICIEV)*, pages 809–812. IEEE, 2016.

[56] Texas Instruments. INA128 Precision Instrumentation Amplifier. `https://www.ti.com/lit/ds/symlink/ina128.pdf`, 2015.

[57] Bruce B. Winter and John G. Webster. Driven-right-leg circuit design. *IEEE Transactions on Biomedical Engineering*, BME-30(1):62–66, 1983.

[58] Yago Antonio do Prado Bertagna. Desenvolvimento de ecg de baixo custo: prototipagem, caracterização elétrica e otimização de layout para facilitar a manutenção. 2024.

[59] Texas Instruments. Current sense circuit collection — making sense of current. Technical Report SBFA012, Texas Instruments, 2006. Application Report.

[60] Andreas Sprenger, Birte Neppert, Sabine Köster, Steffen Gais, Detlef Kömpf, Christoph Helmchen, and Hubert Kimmig. Long-term eye movement recordings with a scleral search coil-eyelid protection device allows new applications. *Journal of neuroscience methods*, 170(2):305–309, 2008.

Apéndice A

Tabla A.1: Términos y variables utilizados en el procesamiento de la señal EOG.

Términos y variables	**Descripción**	**Cómo se obtienen**
Información de interés	Datos adquiridos del EOG siguiendo una rutina mostrada en el campo de visión, fuera de esa rutina, los datos no son de interés.	Realizar rutina de seguimiento ocular estando conectado al sistema de adquisición.
Inicio y fin	Permite recortar la señal desde el primer dato (inicio) hasta el último que se establezca (fin).	Esta se establece observando la información de interés, en donde se establecerá el fin hasta el punto en el que se querrá recortar la señal.
Media móvil (k)	Ventana de tamaño "k" utilizada para promediar datos actuales en base a datos pasados promediados.	Se ajustó a 25 por prueba y error, este valor puede ser mayor o menor en base al ruido de la señal.
Selección de datos en la gráfica "Recortada suavizada"	Diferencia cuando se supera el valor de "Umbral" y cuando se está por debajo de éste nuevamente.	Con el cursor, colocar un marcador en el siguiente primer dato cuando es superado el "Umbral" y otro cuando se está por debajo del mismo en el pico actual en donde se cumplió la condición. Calcular la diferencia en esos dos marcadores y sumar aproximadamente 50 datos más.
Datos offset (n)	Cantidad de datos a utilizar para el cálculo del voltaje offset (promediar desde el dato 1 hasta el n).	Observar la gráfica de señal cruda y seleccionar un valor de datos entre 10 y 1000. El valor será el índice cuando la señal AC sea estable o no se estén realizando movimientos oculares antes de que ocurra el primero de estos.
Umbral	Valor mínimo para detectar un pico positivo o negativo en la señal.	Establecer en base a los picos más grandes de la señal (positivos y negativos).
Distancia entre datos (índices)	El valor de la diferencia entre el índice cuando se supera el umbral en los picos más grandes hasta que se está por debajo de éste nuevamente.	Tomando como ejemplo la Figura 21, observar con el cursor el primer punto desde que se superó el umbral hasta el primer valor debajo del mismo, a ese aproximado de la diferencia, sumar un valor de 50.
Último dato de interés	El índice aproximado al último dato que resulte perteneciente a la rutina de seguimiento ocular.	Observar la señal y seleccionar un valor aproximado al último dato de interés perteneciente a la rutina de seguimiento ocular.

Apéndice B

```
classdef mainx < matlab.apps.AppBase
% Properties that correspond to app components
properties (Access = public)
UIFigure                          matlab.ui.Figure
DatosparacalibracionobtenidosLamp  matlab.ui.control.Lamp
DatosparacalibracionobtenidosLampLabel  matlab.ui.control.Label
AnchoLabel                        matlab.ui.control.Label
AltoLabel                         matlab.ui.control.Label
valorAncho                        matlab.ui.control.Label
valorAlto                         matlab.ui.control.Label
DimensionesdelmonitorLabel        matlab.ui.control.Label
ProcesoButtonGroup                matlab.ui.container.ButtonGroup
AproximacinDCenVypixelesButton  matlab.ui.control.RadioButton
CalibracinMximosyMnimosButton   matlab.ui.control.RadioButton
LimpiarButton                     matlab.ui.control.Button
MaxResult                         matlab.ui.control.Label
MinResult                         matlab.ui.control.Label
MaxLabel                          matlab.ui.control.Label
MinLabel                          matlab.ui.control.Label
PromediosLabel                    matlab.ui.control.Label
CondicinparadeteccinPanel         matlab.ui.container.Panel
PromediarpicosmnimosdetectadosCheckBox  matlab.ui.control.CheckBox
DistanciaentredatosparanuevadeteccinSpinner  matlab.ui.control.Spinner
DistanciaentredatosparanuevadeteccinSpinnerLabel  matlab.ui.control.Label
UmbralSpinner                     matlab.ui.control.Spinner
UmbralSpinnerLabel                matlab.ui.control.Label
```

```
ResultadoButton                 matlab.ui.control.Button
RecotarSealaRegiondeIntersyclcularVoffsetconlosprimerosPanel
matlab.ui.container.Panel
RecortarysuavizarButton         matlab.ui.control.Button
DatossuavizadokSpinner          matlab.ui.control.Spinner
DatossuavizadokSpinnerLabel     matlab.ui.control.Label
DatosoffsetNSpinner             matlab.ui.control.Spinner
DatosoffsetLabel                matlab.ui.control.Label
FinSpinner                      matlab.ui.control.Spinner
FinSpinnerLabel                 matlab.ui.control.Label
InicioSpinner                   matlab.ui.control.Spinner
InicioSpinnerLabel              matlab.ui.control.Label
CargarsealparacalibrarMximosyMnimosPanel  matlab.ui.container.Panel
BuscarArchivoButton             matlab.ui.control.Button
ArchivoEncontrado               matlab.ui.control.Label
CargarButton                    matlab.ui.control.Button
ReconstruccinDCyestimacinenpixelesLabel  matlab.ui.control.Label
TabGroup                        matlab.ui.container.TabGroup
CrudaTab                        matlab.ui.container.Tab
UIAxes1DC                       matlab.ui.control.UIAxes
UIAxes                          matlab.ui.control.UIAxes
RecortadaTab                    matlab.ui.container.Tab
posSlider_2Label                matlab.ui.control.Label
valorPequenoPermitido           matlab.ui.control.Label
posSlider                       matlab.ui.control.Slider
SiguienteButton                 matlab.ui.control.Button
posLowerLabel                   matlab.ui.control.Label
UIAxes4                         matlab.ui.control.UIAxes
UIAxes2                         matlab.ui.control.UIAxes
ResultadoTab                    matlab.ui.container.Tab
UIAxes3                         matlab.ui.control.UIAxes
SealCruda2Tab                   matlab.ui.container.Tab
UIAxes2DC                       matlab.ui.control.UIAxes
UIAxes_2                        matlab.ui.control.UIAxes
Suavizaryrecortar2Tab           matlab.ui.container.Tab
UIAxes_4                        matlab.ui.control.UIAxes
```

```
UIAxes_3                          matlab.ui.control.UIAxes
ReconstruccionTab                 matlab.ui.container.Tab
GuardarInformacindeGrficasPanel   matlab.ui.container.Panel
Nombredearchivoaguardar           matlab.ui.control.EditField
NombredearchivoEditFieldLabel     matlab.ui.control.Label
csvLabel                          matlab.ui.control.Label
GuardarButton                     matlab.ui.control.Button
UIAxes3_4                         matlab.ui.control.UIAxes
UIAxes3_3                         matlab.ui.control.UIAxes
    end
```

Apéndice C

```
% Public properties that correspond to the Simulink model
properties (Access = public, Transient)
Simulation
end
properties (Access = public)
% CREACION DE VARIABLES E INICIALIZACIÓN DE LAS MISMAS
Property
state = 'A'; % Variable para cambiar de casos (Transición entre botones)
Ruta = ''; % Guardar la ruta de un archivo a cargar
data = []; % Almacenamiento de la señal en arreglos
data2 = []; % en cruda o procesada con suavizado
data3 = []; %
dataDC = [];  % Arreglo de valores en DC en el archivo seleccionado
inicioSpinner = 1; % Comienzo de la señal
finSpinner = 2000; % En donde debería terminar la señal (recortar)
no_invertida = []; % Almacenar el arreglo de la señal sin invertirla
invertida = []; % Almacenar la señal AC invertida
offset = 0; % Establecer cantidad de datos para calcular una media offset
offsetV = 0; % Voltaje offset calculado, este se restará a la señal debido
%a que no esta montada en la línea base 0
pksGeneral = []; % Arreglo que almacena picos positivos y negativos de la
%señal AC suavizada
r = 0;
segmentoData = []; % Almacenará la señal filtrada
k = 25; % Datos para aplicar suavizado de media móvil
pksPositivo = []; % Almacenar picos positivos
```

```
pksNegativo = []; % Almacenar picos negativos
meanpksPositivo = 0; % Calculo del promedio de los picos positivos que
%superaron el umbral +
meanpksNegativo = 0; % Cálculo del promedio de los picos negativos que
%superaron el umbral -
calibrada = false; % Variable booleana para comprobar si se realizó la
%primer calibracion (t = c, f = nc)
val_min = 0; % Variable que almacena el dato del slider umbral ubicado
%en el Tab "Recortada"
% Implementadas para realizar la aproximación a DC utilizando los
% picos en AC
suma = 0;
saveS = 0;
ult = [0,0];
recData = [];
% Variables utilizadas en la conversión a píxeles
dimensiones_monitor;
alto;
ancho;
coordPixel;
% Variables utilizadas para poder nombrar el cómo se guardarán los
%resultados en un archivo del tipo ".csv"
Narchivo = '';
NarchivoCompleto = '';
% Variables asignadas a la función de comprobación de pico mínimo
abs_suavizada;
pksEvaluar;
data4;
ResultadoPMinimo; % Variable utilizada para almacenar el valor del
% pico más pequeño en el modo de uso 2
end
```

Apéndice D

```
function CARGAR(app)
% Desactivar Botón de Carga
app.CargarButton.Enable = 'off';
drawnow; % Imprimir el valor más pequeño
archivo = app.ArchivoEncontrado.Text; % Obtener la ruta completa
if exist(archivo, 'file') ~= 2
disp('Error: No se encontró el archivo seleccionado.');
return;
end
% Cargar archivos en formato CSV
DC_AC = readtable(archivo);
% % Guardar la nueva tabla a un archivo temporal para la
% conversión a formato tipo MAT (.mat)
ruta_temporal = fullfile(app.Ruta, 'temporal.csv');
writetable(DC_AC, ruta_temporal);
DC_AC = readmatrix(ruta_temporal);
% Guardado de datos en MAT
save('archivo_AC_DC.mat', 'DC_AC');
datos = load('archivo_AC_DC.mat');
app.data = datos.DC_AC(:,2)';
app.dataDC = datos.DC_AC(:,1)';
% Activar los spinners para controlar el inicio y fin de la
% señal.
app.InicioSpinner.Enable = 'off'; % Dejar desactivado el spinner.
app.FinSpinner.Enable = 'on'; % Activa el spinner "Fin"
app.DatosoffsetNSpinner.Enable = 'on'; % Activar el spinner
```

```
% que controla la cantidad de datos a usar para el cálculo
% del voltaje offset.
app.RecortarysuavizarButton.Enable = 'on'; % Avtivar el botón
% para confirmar parámetros establecidos.
app.DatossuavizadokSpinner.Enable = 'on'; % Activar el control
% encargado de modificar el tamaño de la ventana para el
% suavizado por media móvil.
end
```

Apéndice E

```
function RECORTAR(app)
app.offsetV = mean(app.data(app.inicioSpinner:app.offset));
app.data2 = app.data(app.inicioSpinner:app.finSpinner);
app.data3 = smooth(app.data2, app.k) - app.offsetV;
app.data4 = smooth(app.data2, app.k) - app.offsetV;
% Activar controles del panel para obtener datos de calibración
app.UmbralSpinner.Enable = 'on';
app.ResultadoButton.Enable = 'on';
app.DistanciaentredatosparanuevadeteccinSpinner.Enable = 'on';
% Desactivar controles de recortar señal
app.InicioSpinner.Enable = 'off';
app.FinSpinner.Enable = 'off';
app.DatosoffsetNSpinner.Enable = 'off';
app.RecortarysuavizarButton.Enable  = 'off';
app.DatossuavizadokSpinner.Enable = 'off';
app.PromediarpicosmnimosdetectadosCheckBox.Enable = 'on';
if app.AproximacinDCenVypixelesButton.Value == true
app.PromediarpicosmnimosdetectadosCheckBox.Enable = 'off';
end
end
```

Apéndice F

```
function RESULTADO(app)
app.ResultadoButton.Enable = 'off';
app.PromediarpicosmnimosdetectadosCheckBox.Enable = 'off';
if app.CalibracinMximosyMnimosButton.Value == true
app.posSlider.Enable = 'on';
end
drawnow;
% Comportamiento del estado C
app.no_invertida = app.data2; % Señal original recortada
%sin invertir
app.invertida = - app.data2; % Señal original recortada
%invertida para detección de picos negativos
%% Establecer el umbral a utilizar con condiciones
% Si ya se realizó la primer calibración y se marcó la casilla
% que se encuentra en el grupo de "Condición para detección"
% entonces se establece el pico mínimo permitido para evitar
% conflictos con picos indeseados
if app.calibrada == true && app.PromediarpicosmnimosdetectadosCheckBox.Value == true
minPeakHeight = app.posSlider.Value;
%if app.calibrada == true && app.
Promediarpicosmnimosdetectados
%CheckBox.Value == true
minPeakHeight = app.posSlider.Value;
% Primer umbral usado en caso que no exista una calibración
% previa y el checkbox no este marcado
else
```

```
minPeakHeight = app.UmbralSpinner.Value;
end
% Distancia entre datos a esperar para nueva detección
minPeakDistance = app.Distanciaentredatos
paranuevadeteccinSpinner.Value;
%% Inicializar vectores para almacenar la señal procesada
N = length(app.data2); % Longitud total de la señal
app.segmentoData = zeros(1, N); % Almacenará la señal filtrada
% Inicializar variables para picos positivos y negativos
app.pksGeneral = zeros(1, N); % Cuando es detectado un pico es
%almacenado en esta variable
count = minPeakDistance + 1; % Establecer un contador para
%interpretar cuando se detectó un pico
% y si ya pasó cierta cantidad de datos para poder detectar
% otro
media_offset = app.DatosoffsetNSpinner.Value;
% En esta variable se almacena el valor a utilizar
% para calcular el offset de la señal, de modo que se guardan
% cierta cantidad de datos para promediar, posteriormente, el
%resultado es restado a la señal suavizada.
%% CICLO PARA CALCULAR LOS PICOS Y RECONSTRUIR LA SEÑAL APROXIMADA A DC
for i = 1:N
% Inicio del ciclo para cálculo de picos y reconstrucción
%dataR = app.data3(1:i);
% En cada iteración se agrega un
% valor al arreglo, dicho valor es el que se encuentra en
% el índice "i" de la señal suavizada
if i > media_offset + 1 % No se entra a la detección de los picos
% hasta haber superado al valor establecido en número
% de datos para calcular offset.
%% Buscar picos positivos en el segmento actual de datos
dataS(i) = app.data3(i); % Almacenar el valor de
% de la señal suavizada AC en determinada muestra "i"
app.segmentoData = dataS(1:i); % Conforme se van almacenando
% valores, se va creando un arreglo de datos llamado
% segmentoData
```

```
if i >= minPeakDistance + 2 % De igual forma, si el valor de
% "i" es mayor que minPeakDistance, comienza la
% detección de picos, si esta condición no se
% cumple se mantendrá el valor en el ultimo pico
% detectado
if abs(app.segmentoData(i)) >= minPeakHeight % Se
% crea una condición para saber si el valor
% absoluto de la señal al momento actual supera
% el umbral establecido, ya sea el primer
% umbral utilizado para calibración y
% evaluación en caso de existir dos señales
% (una para calibrar y otra para evaluar) o
% bien, si se supera el segundo umbral en caso
% de hacer calibración y evaluación con una
% señal que contenga una señal con distintas
% amplitudes eligiendo los picos mas altos para
% el primer umbral como calibración con el fin
% de obtener los valores mas altos de la señal
% y delimitar el margen horizontal de visión y el
% segundo umbral para permitir los picos
% mínimos
%% A grandes rasgos, se buscan los picos en la
% señal suavizada hasta el momento actual en un
% rango de datos desde "minPeakDistance" hasta
% "i" al cumplirse la condición de if abs(app.
% segmentoData(i)) >= minPeakHeight indicando
% que se superó el umbral de forma positiva o
% negativa
[pks, ~] = findpeaks(app.segmentoData(i-minPeakDistance:i), 'MinPeakHeight', minPeakHeight);% form
if ~isempty(pks) && count > minPeakDistance
count = 1; % Se reinicia el contador a 1
% para realizar una nueva detección
% Almacenar los últimos picos detectados
pks1 = findpeaks(dataR(i-50:end));
app.pksGeneral(i) = (max(pks1));
% Se busca el pico más grande almacenado
```

```
% dentro al terminar la detección desde que
% se superó el umbral y transcurrió la cantidad
% de datos establecida en "minPeakDistance".
%% Sección usada para aproximar a píxeles
% los picos obtenidos en Voltaje
if app.calibrada == true && app.AproximacinDCenVypixelesButton.Value == true
app.saveS = app.suma;
app.ult(1) = app.pksGeneral(i);
app.ult(2) = app.saveS;
end
end
% Buscar picos negativos (invierte el segmento de datos)
[pks, ~] = findpeaks(-app.segmentoData(i-minPeakDistance:i),
'MinPeakHeight', minPeakHeight); % Se invierte
% la señal desde que se van creando datos hasta
% llegar a "i", (dato actual) de modo que se
% puedan obtener los picos negativos
% Si existen datos en pks (es decir, que ya se
% detectaron picos) y el contador supera a
% "minPeakDistance" se procede a buscar el pico
% más alto dentro de los picos almacenados en
% una sola detección (se hizo porque se
% presentaban varios picos en una sola
% detección a causa del ruido)
if ~isempty(pks) && count > minPeakDistance
count = 1; % Se reinicia el contador a 1
% para realizar una nueva detección
pks1 = findpeaks(-dataR(i-50:end));
% Almacenar los últimos picos negativos
% detectados y convertirlos a valores negativos
app.pksGeneral(i) = -(max(pks1)); % Buscar
% el pico mas alto dentro de los que se
% detectaron una vez se superó el umbral y
% pasaron cierta cantidad de datos determinada
% por minPeakDistance datos desde que el
% umbral fue superado.
```

```
%% Sección usada para aproximar a píxeles
if app.calibrada == true
app.saveS = app.suma;
app.ult(1) = app.pksGeneral(i);
app.ult(2) = app.saveS;
end
end
end
%%  Aproximación a píxeles calculada y aplicada
if app.calibrada == true && app.AproximacinDCenVypixelesButton.Value == true
app.suma = app.ult(1) + app.ult(2);
app.recData(i) = app.suma;
app.coordPixel(i) =round(((app.recData(i) - app.meanpksNegativo) /
(app.meanpksPositivo - app.meanpksNegativo)) * (app.ancho));
end
%% Aumentar el contador en 1 hasta llegar a ser
% igual o menor a minPeakDistance
if count <= minPeakDistance
count = count + 1;
end
end
end
end
end
```

Apéndice G

```
%% FUNCION APLICADA PARA CALCULAR EL PICO MAS PEQUEÑO PERMITIDO
% (UTIL PARA REDUCIR PROBLEMAS CON PICOS CONFUNDIBLES CON RUIDO
% O FUERA DE INTERES)
function comprobar_picominimo(app)
% app.CalibracinMximosyMnimosButton.Value == true &&
if app.PromediarpicosmnimosdetectadosCheckBox.Value == true && app.calibrada ==
true
cla(app.UIAxes4);
%if app.PromediarpicosmnimosdetectadosCheckBox.Value == true &&
%app.calibrada == true
cla(app.UIAxes4);
% Limpiar grafico
drawnow;
% Y mostrar limpieza al instante
% Cambiar propiedades de controles y graficar la señal AC
% suavizada con valor absoluto.
app.ResultadoButton.Enable = 'off';
app.SiguienteButton.Enable = 'on';
app.PromediarpicosmnimosdetectadosCheckBox.Enable = 'off';
app.abs_suavizada = abs(app.data4);
plot(app.UIAxes4, app.abs_suavizada);
drawnow;
end
end
```

Apéndice H

```
function RESULTADOmp(app)
% Cambiar propiedades del botón de resultado para detección de
% picos mas pequeños
app.SiguienteButton.Enable = 'off';
app.posSlider.Enable = 'off';
drawnow;
end
```

Apéndice I

```
%% Guardar el resultado de lo mostrado en el Tab Reconstrucción
function guardar(app)
if app.PromediarpicosmnimosdetectadosCheckBox.Value == false
app.Narchivo = string(app.Narchivo);
app.NarchivoCompleto = strcat(app.Narchivo, app.csvLabel.Text);
FolderGuardar = 'C:\AC2DC_EOG\application\
Conjuntos_senales\Aproximaciones\';
datosAguardar1 = transpose(app.recData);
datosAguardar2 = transpose(app.coordPixel);
datos_combinados = [datosAguardar1, datosAguardar2];
PathCompleto = fullfile(FolderGuardar, app.NarchivoCompleto);
% Guardar los arreglos en el archivo CSV
writematrix(datos_combinados, PathCompleto);
else
app.Narchivo = string(app.Narchivo);
app.NarchivoCompleto = strcat(app.Narchivo, app.csvLabel.Text);
FolderGuardar = 'C:\AC2DC_EOG\application
\Conjuntos_senales_m2\Aproximaciones\';
datosAguardar1 = transpose(app.recData);
datosAguardar2 = transpose(app.coordPixel);
datos_combinados = [datosAguardar1, datosAguardar2];
PathCompleto = fullfile(FolderGuardar, app.NarchivoCompleto);
% Guardar los arreglos en el archivo CSV
writematrix(datos_combinados, PathCompleto);
end
end
```

Apéndice J

```
%% Regresar al inicio del proceso una vez calculado la primer calibración
function regreso(app)
app.DistanciaentredatosparanuevadeteccinSpinner.Enable = "off";
app.UmbralSpinner.Enable = 'off';
app.ResultadoButton.Enable = 'off';
app.CargarButton.Enable = 'on';
end
```

Apéndice K

```
%% Limpiar variables (Reiniciar todas)
function LIMPIARVAR(app)
app.data = [];
app.data2 = [];
app.data3 = [];
app.no_invertida = [];
app.invertida = [];
app.segmentoData = [];
app.pksGeneral = [];
app.r = 0;
app.offsetV = 0;
app.pksPositivo = 0;
app.pksNegativo = 0;
app.suma = 0;
app.saveS = 0;
app.ult = [0,0];
app.coordPixel = [];
app.recData = [];
% Habilitar botón de carga
app.CargarButton.Enable = 'on';
% Desactivar controles de recortar señal
app.InicioSpinner.Enable = 'off';
app.DatossuavizadokSpinner.Enable = 'off';
app.FinSpinner.Enable = 'off';
app.DatosoffsetNSpinner.Enable = 'off';
app.RecortarysuavizarButton.Enable = 'off';
```

```
app.UmbralSpinner.Enable = 'off';
app.ResultadoButton.Enable = 'off';
app.DistanciaentredatosparanuevadeteccinSpinner.Enable = 'off';
app.AproximacinDCenVypixelesButton.Enable = 'off';
app.CalibracinMximosyMnimosButton.Enable = 'on';
app.SiguienteButton.Enable = 'off';
app.posSlider.Enable = 'off';
app.PromediarpicosmnimosdetectadosCheckBox.Enable = 'off';
app.posLowerLabel.Text = sprintf('Umbral Pico Mínimo:...');
app.valorPequenoPermitido.Text = sprintf('Pico más pequeño:...');
end
```

Apéndice L

```
%% Limpiar variables almacenadoras de los picos, reinicia casillas que muestran
% valores mínimo y maximo y sus respectivas variables
function LIMPIARMAXMIN(app)
app.pksPositivo = [];
app.pksNegativo = [];
app.meanpksPositivo = 0;
app.meanpksNegativo = 0;
app.MaxResult.Text = ".......";
app.MinResult.Text = ".......";
app.calibrada = false;
app.DatosparacalibracionobtenidosLamp.Color = 'red';
app.ProcesoButtonGroup.SelectedObject = app.
CalibracinMximosyMnimosButton;
app.PromediarpicosmnimosdetectadosCheckBox.Value = false;
end
```

Apéndice M

```
%% Limpia todos los graficos
function LIMPIARGRAPH(app)
cla(app.UIAxes); % Señal sin procesar en forma de AC (Calibración)
cla(app.UIAxes1DC); % % Señal sin procesar en forma DC (Calibración)
cla(app.UIAxes_2); % Señal sin procesar en forma de AC (Aproximación)
cla(app.UIAxes2DC); % Señal cruda sin procesar en DC (Aproximación)
cla(app.UIAxes2); % Muestra la señal recortada (Calibración)
cla(app.UIAxes3); % Muestra los picos detectados en
la señal (Calibración)
% suavizada y recortada
cla(app.UIAxes4); % Recortada y suavizada (Calibración)
cla(app.UIAxes_3); % Señal recortada (Aproximación)
cla(app.UIAxes_4); % Señal recortada y suavizada (Aproximación)
cla(app.UIAxes3_3); % Aproximación a DC
cla(app.UIAxes3_4); % Gráfico "Conversión a píxeles"
end
```

Apéndice N

```
%% GRAFICAR LA INFORMACION CALCULADA Y ALMACENADA EN ARREGLOS
function plotOriginal(app)
plot(app.UIAxes, app.data);
end
function plotRecortado(app)
plot(app.UIAxes2, app.data2);
end
function plotRecortadoS(app)
plot(app.UIAxes4, app.data3);
end
function plotOriginal2(app)
plot(app.UIAxes_2, app.data);
end
function plotRecortado2(app)
plot(app.UIAxes_3, app.data2);
end
function plotRecortado2S(app)
plot(app.UIAxes_4, app.data3);
end
function plotReconstruccionV(app)
plot(app.UIAxes3_3, app.recData);
end
function plotReconstruccionP(app)
plot(app.UIAxes3_4, app.coordPixel);
end
function plotCruda1DC(app)
```

```
plot(app.UIAxes1DC, app.dataDC)
end
function plotCruda2DC(app)
plot(app.UIAxes2DC, app.dataDC)
end
```

Apéndice Ñ

```
%% MOSTRAR LOS PICOS DETECTADOS EN EL GRAFICO DEL TAB "RESULTADO"
function plotResultado(app)
cla(app.UIAxes3); % Limpiar la gráfica UIAxes3
inicio = app.InicioSpinner.Value;
fin = app.FinSpinner.Value;
app.r = fin - inicio + 1;
% + 1 para incluir el último punto en el rango
% Asegurarse de que el índice final no exceda el tamaño del vector de picos
if fin <= length(app.pksGeneral)
plot(app.UIAxes3, (1:app.r), app.pksGeneral(inicio:fin));
else
plot(app.UIAxes3, (1:length(app.pksGeneral)), app.pksGeneral);
end
end
```

Apéndice O

```
%% CALCULAR PROMEDIO DE PICOS Y PROMEDIO DE VALLES REPRESENTATIVOS
% O PERTENECIENTES A SACADICOS DE CENTRO A EXTREMO MAXIMO DERECHO O
% CENTRO A EXTREMO MAXIMO IZQUIERDO.
function calculoMinMax(app)
% Separar en variables los picos positivos y los negativos
app.pksPositivo = app.pksGeneral(app.pksGeneral > 0);
app.pksNegativo = app.pksGeneral(app.pksGeneral < 0);
% Calcular el promedio por separado
app.meanpksPositivo = mean(app.pksPositivo);
app.meanpksNegativo = mean(app.pksNegativo);
% Mostrar el resultado de ambos promedios en dos Label
app.MaxResult.Text = num2str(app.meanpksPositivo);
app.MinResult.Text = num2str(app.meanpksNegativo);
if app.meanpksNegativo ~= 0 && app.meanpksPositivo ~= 0
app.AproximacinDCenVypixelesButton.Enable = 'on';
app.CalibracinMximosyMnimosButton.Enable = 'off';
app.calibrada = true;
app.DatosparacalibracionobtenidosLamp.Color = 'green';
           else
app.AproximacinDCenVypixelesButton.Enable = 'off';
end
end
```

Apéndice P

```
% Código en MATLAB
%% SECCION EN DONDE SE EJECUTAN LAS FUNCIONES PREVIAMENTE CREADAS
% EN BASE A UN SELECTOR POR CASOS
% LOS CASOS DEPENDEN DE LA SELECCION DE BOTONES MOSTRADOS EN EL
% PANEL FRONTAL
function stateTransition(app, newState)
app.state = newState;
switch app.state
case 'A'
if app.ArchivoEncontrado.Text ~= "" &&
app.CalibracinMximosyMnimosButton.Value == true
app.LIMPIARVAR();
if app.calibrada == false
app.LIMPIARGRAPH();
end
app.CARGAR();
app.plotOriginal();
app.plotCruda1DC();
elseif app.ArchivoEncontrado.Text ~= "" &&
app.AproximacinDCenVypixelesButton.Value == true
app.LIMPIARVAR();
if app.calibrada == false
app.LIMPIARGRAPH();
end
app.CARGAR();
app.plotOriginal2();
```

```
app.plotCruda2DC();
end
case 'B'
app.RECORTAR();
if app.calibrada == false && app.CalibracinMximosyMnimosButton.Value == true
app.plotRecortadoS();
app.plotRecortado();
elseif app.calibrada == true && app.AproximacinDCenVypixelesButton.Value == true
% En caso de tener los datos de calibración, se procede a
% apagar los controles de panel de resultado
% excepto el boton
app.UmbralSpinner.Enable = 'off';
app.ResultadoButton.Enable = 'on';
app.DistanciaentredatosparanuevadeteccinSpinner.Enable = 'off';
drawnow;
app.plotRecortado2S();
app.plotRecortado2();
end
case 'C'
app.DistanciaentredatosparanuevadeteccinSpinner.Enable = 'off';
app.UmbralSpinner.Enable = 'off';
app.LimpiarButton.Enable = 'off';
drawnow;
app.RESULTADO();
app.plotResultado();
if app.CalibracinMximosyMnimosButton.Value == true
app.calculoMinMax();
elseif app.AproximacinDCenVypixelesButton.Value == true && app.calibrada == true
app.plotReconstruccionV();
app.plotReconstruccionP();
app.ResultadoButton.Enable = 'off';
end
app.LimpiarButton.Enable = 'on';
app.SiguienteButton.Enable = 'on';
app.ArchivoEncontrado.Text = "";
app.comprobar_picominimo();
```

```
case 'D'
app.LIMPIARVAR();
app.LIMPIARMAXMIN();
app.LIMPIARGRAPH();
app.TabGroup.SelectedTab = app.CrudaTab;
case 'E'
app.guardar();
case 'F'
app.regreso();
case 'G'
app.RESULTADOmp();
end
end
end
```

Apéndice Q

```
% Callbacks that handle component events
methods (Access = private)
% Code that executes after component creation
function startupFcn(app)
app.offset = app.DatosoffsetNSpinner.Value;
% Habilitar botón de carga
app.CargarButton.Enable = 'on';
% Deshabilitar Botones
app.RecortarysuavizarButton.Enable = 'off';
app.ResultadoButton.Enable = 'off';
app.InicioSpinner.Enable = 'off';
app.FinSpinner.Enable = 'off';
app.DistanciaentredatosparanuevadeteccinSpinner.Enable = 'off';
app.UmbralSpinner.Enable = 'off';
app.DatosoffsetNSpinner.Enable = 'off';
app.DatossuavizadokSpinner.Enable = 'off';
app.DatosparacalibracionobtenidosLamp.Color = 'red';
app.ProcesoButtonGroup.SelectedObject = app.CalibracinMximosyMnimosButton;
app.AproximacinDCenVypixelesButton.Enable = 'off';
app.dimensiones_monitor = get(0, 'ScreenSize');
app.ancho = app.dimensiones_monitor(3);  % Ancho del monitor
app.alto = app.dimensiones_monitor(4); % Alto del monitor
app.valorAncho.Text = num2str(app.ancho);
app.ancho = app.dimensiones_monitor(3) - 20;
app.valorAlto.Text = num2str(app.alto);
% Configurar límites y valores iniciales de los sliders
```

```
app.posSlider.Limits = [0.001, 1];
app.posSlider.Value = 0.001; % Valores iniciales de los cursores
app.SiguienteButton.Enable = 'off';
app.posLowerLabel.Text = sprintf('Umbral Pico Mínimo:...');
app.valorPequenoPermitido.Text = sprintf('Pico más pequeño:...');
end
% Button pushed function: ResultadoButton
function ResultadoButtonPushed(app, event)
app.stateTransition('C');
if app.calibrada == true && app.AproximacinDCenVypixelesButton.Value == true
app.TabGroup.SelectedTab = app.ReconstruccionTab;
else
app.TabGroup.SelectedTab = app.ResultadoTab;
if app.PromediarpicosmnimosdetectadosCheckBox.Value == true
app.TabGroup.SelectedTab = app.RecortadaTab;
app.posSlider.Limits = [0.001, app.meanpksPositivo*1.1];
app.posSlider.Value = app.meanpksPositivo * 0.9;
app.posLowerLabel.Text = sprintf('Umbral Pico Mínimo: %.3f',
app.meanpksPositivo*0.9);
end
end
end
% Button pushed function: RecortarysuavizarButton
function RecortarysuavizarButtonPushed(app, event)
app.stateTransition('B');
if app.calibrada == true && app.AproximacinDCenVypixelesButton.Value == true
app.TabGroup.SelectedTab = app.Suavizaryrecortar2Tab;
else
app.TabGroup.SelectedTab = app.RecortadaTab;
end
end
% Button pushed function: CargarButton
function CargarButtonPushed(app, event)
app.stateTransition('A');
if app.calibrada == true && app.AproximacinDCenVypixelesButton.Value == true
app.TabGroup.SelectedTab = app.SealCruda2Tab;
```

```
else
app.TabGroup.SelectedTab = app.CrudaTab;
end
end
% Value changed function: InicioSpinner
function InicioSpinnerValueChanged(app, event)
app.inicioSpinner = app.InicioSpinner.Value;
end
% Value changed function: FinSpinner
function FinSpinnerValueChanged(app, event)
app.finSpinner = app.FinSpinner.Value;
end
% Value changed function: DatosoffsetNSpinner
function DatosoffsetNSpinnerValueChanged(app,event)
app.offset = app.DatosoffsetNSpinner.Value;
end
% Value changed function: %DistanciaentredatosparanuevadeteccinSpinner
function DistanciaentredatosparanuevadeteccinSpinnerValueChanged(app, event)
end
% Value changed function: UmbralSpinner
function UmbralSpinnerValueChanged(app, event)
end
% Button pushed function: LimpiarButton
function LimpiarButtonPushed(app, event)
app.stateTransition('D');
end
% Selection changed function: ProcesoButtonGroup
function ProcesoButtonGroupSelectionChanged(app, event)
app.stateTransition('F');
selectedButton = app.ProcesoButtonGroup.SelectedObject;
end
% Button pushed function: BuscarArchivoButton
function BuscarArchivoButtonPushed(app, event)
[filename, pathname] = uigetfile({'*.csv','Archivos soportados (*.csv)'}, ...
'Seleccione un archivo');
if isequal(filename,0)
```

```
disp('No se seleccionó ningún archivo.');
else
fullpath = fullfile(pathname, filename);
app.ArchivoEncontrado.Text = fullpath;  % Guardar la ruta completa
app.Ruta = pathname;  % Guardar también el directorio de trabajo
end
end
% Value changed function: DatossuavizadokSpinner
function DatossuavizadokSpinnerValueChanged(app, event)
app.k = app.DatossuavizadokSpinner.Value;
end
% Button pushed function: GuardarButton
function GuardarButtonPushed(app, event)
app.stateTransition('E');
end
% Value changed function: Nombredearchivoaguardar
function NombredearchivoaguardarValueChanged(app, event)
app.Narchivo = app.Nombredearchivoaguardar.Value;
end
% Value changed function: posSlider
function posSliderValueChanged(app, event)
app.val_min = app.posSlider.Value;
app.posLowerLabel.Text = sprintf('Umbral Pico Mínimo: %.3f', app.val_min);
% Validar que abs_suavizada es un vector
app.abs_suavizada = app.abs_suavizada(:);
% Detectar picos que superen el valor del umbral del slider (al
% valor que se ajustó)
validIndices = app.abs_suavizada >= app.val_min;
filteredSignal = app.abs_suavizada(validIndices);
% Congfigurar los parámetros para detección de picos
minPeakHeight = app.posSlider.Value; % Umbral mínimo de amplitud permitido
minPeakProminence = 0.01; % Prominencia mínima del pico
% Encontrar picos dentro del rango con parámetros (Señal
% completa)
[app.pksEvaluar, ~] = findpeaks(filteredSignal, ...
'MinPeakHeight', minPeakHeight, ...
```

```
'MinPeakProminence', minPeakProminence);
%% Calcular el pico más pequeño permitido
if isempty(app.pksEvaluar)
% Si no hay picos, mostrar mensaje
app.valorPequenoPermitido.Text = 'No se detectaron picos';
else
% Si hay picos, mostrar el más pequeño
[app.ResultadoPMinimo, ~] = min(app.pksEvaluar);
app.valorPequenoPermitido.Text = sprintf
('Pico más pequeño: %.3f',app.ResultadoPMinimo);
end
end
% Button pushed function: SiguienteButton
function SiguienteButtonPushed(app, event)
app.stateTransition('G');
end
end
% Component initialization
methods (Access = private)
% Create UIFigure and components
function createComponents(app)
% Create UIFigure and hide until all components are created
app.UIFigure = uifigure('Visible', 'off');
app.UIFigure.Color = [1 1 1];
app.UIFigure.Position = [100 100 1002 598];
app.UIFigure.Name = 'MATLAB App';
app.UIFigure.Scrollable = 'on';
% Create TabGroup
app.TabGroup = uitabgroup(app.UIFigure);
app.TabGroup.Position = [456 10 536 534];
% Create CrudaTab
app.CrudaTab = uitab(app.TabGroup);
app.CrudaTab.Title = 'Cruda';
% Create UIAxes
app.UIAxes = uiaxes(app.CrudaTab);
title(app.UIAxes, 'Señal Cruda AC')
```

```
xlabel(app.UIAxes, 'n')
ylabel(app.UIAxes, 'Amplitud (V)')
zlabel(app.UIAxes, 'Z')
app.UIAxes.XGrid = 'on';
app.UIAxes.YGrid = 'on';
app.UIAxes.Position = [8 293 519 212];
% Create UIAxes1DC
app.UIAxes1DC = uiaxes(app.CrudaTab);
title(app.UIAxes1DC, 'Señal Cruda DC')
xlabel(app.UIAxes1DC, 'n')
ylabel(app.UIAxes1DC, 'Amplitud (V)')
zlabel(app.UIAxes1DC, 'Z')
app.UIAxes1DC.XGrid = 'on';
app.UIAxes1DC.YGrid = 'on';
app.UIAxes1DC.Position = [8 82 519 212];
% Create RecortadaTab
app.RecortadaTab = uitab(app.TabGroup);
app.RecortadaTab.Title = 'Recortada';
% Create UIAxes2
app.UIAxes2 = uiaxes(app.RecortadaTab);
title(app.UIAxes2, 'Recortada')
xlabel(app.UIAxes2, 'n')
ylabel(app.UIAxes2, 'Amplitud (V)')
zlabel(app.UIAxes2, 'Z')
app.UIAxes2.XGrid = 'on';
app.UIAxes2.YGrid = 'on';
app.UIAxes2.Position = [1 297 530 213];
% Create UIAxes4
app.UIAxes4 = uiaxes(app.RecortadaTab);
title(app.UIAxes4, 'Recortada Suavizada')
xlabel(app.UIAxes4, 'n')
ylabel(app.UIAxes4, 'Amplitud (V)')
zlabel(app.UIAxes4, 'Z')
app.UIAxes4.XGrid = 'on';
app.UIAxes4.YGrid = 'on';
app.UIAxes4.Position = [99 75 430 223];
```

```
% Create posLowerLabel
app.posLowerLabel = uilabel(app.RecortadaTab);
app.posLowerLabel.Position = [31 39 236 22];
app.posLowerLabel.Text = '.';
% Create SiguienteButton
app.SiguienteButton = uibutton(app.RecortadaTab, 'push');
app.SiguienteButton.ButtonPushedFcn = createCallbackFcn
(app, @SiguienteButtonPushed, true);
app.SiguienteButton.Position = [431 6 100 23];
app.SiguienteButton.Text = 'Siguiente...';
% Create posSlider
app.posSlider = uislider(app.RecortadaTab);
app.posSlider.Orientation = 'vertical';
app.posSlider.ValueChangedFcn = createCallbackFcn
(app, @posSliderValueChanged, true);
app.posSlider.Position = [40 111 3 168];
% Create valorPequenoPermitido
app.valorPequenoPermitido = uilabel(app.RecortadaTab);
app.valorPequenoPermitido.Position = [303 38 222 22];
app.valorPequenoPermitido.Text = '.';
% Create posSlider_2Label
app.posSlider_2Label = uilabel(app.RecortadaTab);
app.posSlider_2Label.HorizontalAlignment = 'right';
app.posSlider_2Label.Position = [31 283 25 22];
app.posSlider_2Label.Text = 'U.P.M.';
% Create ResultadoTab
app.ResultadoTab = uitab(app.TabGroup);
app.ResultadoTab.Title = 'Resultado';
% Create UIAxes3
app.UIAxes3 = uiaxes(app.ResultadoTab);
title(app.UIAxes3, 'Picos detectados')
xlabel(app.UIAxes3, 'n')
ylabel(app.UIAxes3, 'Amplitud (V)')
zlabel(app.UIAxes3, 'Z')
app.UIAxes3.XGrid = 'on';
app.UIAxes3.YGrid = 'on';
```

```
app.UIAxes3.Position = [2 76 529 434];
% Create SealCruda2Tab
app.SealCruda2Tab = uitab(app.TabGroup);
app.SealCruda2Tab.Title = 'Señal Cruda 2';
% Create UIAxes_2
app.UIAxes_2 = uiaxes(app.SealCruda2Tab);
title(app.UIAxes_2, 'Señal Cruda AC')
xlabel(app.UIAxes_2, 'n')
ylabel(app.UIAxes_2, 'Amplitud (V)')
zlabel(app.UIAxes_2, 'Z')
app.UIAxes_2.XGrid = 'on';
app.UIAxes_2.YGrid = 'on';
app.UIAxes_2.Position = [8 293 519 212];
% Create UIAxes2DC
app.UIAxes2DC = uiaxes(app.SealCruda2Tab);
title(app.UIAxes2DC, 'Señal Cruda DC')
xlabel(app.UIAxes2DC, 'n')
ylabel(app.UIAxes2DC, 'Amplitud (V)')
zlabel(app.UIAxes2DC, 'Z')
app.UIAxes2DC.XGrid = 'on';
app.UIAxes2DC.YGrid = 'on';
app.UIAxes2DC.Position = [8 82 519 212];
% Create Suavizaryrecortar2Tab
app.Suavizaryrecortar2Tab = uitab(app.TabGroup);
app.Suavizaryrecortar2Tab.Title = 'Suavizar y recortar 2';
% Create UIAxes_3
app.UIAxes_3 = uiaxes(app.Suavizaryrecortar2Tab);
title(app.UIAxes_3, 'Señal 2 recortada')
xlabel(app.UIAxes_3, 'n')
ylabel(app.UIAxes_3, 'Amplitud (V)')
zlabel(app.UIAxes_3, 'Z')
app.UIAxes_3.XGrid = 'on';
app.UIAxes_3.YGrid = 'on';
app.UIAxes_3.Position = [8 293 519 212];
% Create UIAxes_4
app.UIAxes_4 = uiaxes(app.Suavizaryrecortar2Tab);
```

```
title(app.UIAxes_4, 'Señal 2 suavizada')
xlabel(app.UIAxes_4, 'n')
ylabel(app.UIAxes_4, 'Amplitud (V)')
zlabel(app.UIAxes_4, 'Z')
app.UIAxes_4.XGrid = 'on';
app.UIAxes_4.YGrid = 'on';
app.UIAxes_4.Position = [8 89 519 203];
% Create ReconstruccionTab
app.ReconstruccionTab = uitab(app.TabGroup);
app.ReconstruccionTab.Title = 'Reconstrucción';
% Create UIAxes3_3
app.UIAxes3_3 = uiaxes(app.ReconstruccionTab);
title(app.UIAxes3_3, 'Aproximación a DC usando señal AC')
xlabel(app.UIAxes3_3, 'n')
ylabel(app.UIAxes3_3, 'Amplitud (V)')
zlabel(app.UIAxes3_3, 'Z')
app.UIAxes3_3.XGrid = 'on';
app.UIAxes3_3.YGrid = 'on';
app.UIAxes3_3.Position = [2 297 529 213];
% Create UIAxes3_4
app.UIAxes3_4 = uiaxes(app.ReconstruccionTab);
title(app.UIAxes3_4, 'Conversión a píxeles')
xlabel(app.UIAxes3_4, 'n')
ylabel(app.UIAxes3_4, 'Coordenada en píxeles - horizontal')
zlabel(app.UIAxes3_4, 'Z')
app.UIAxes3_4.XGrid = 'on';
app.UIAxes3_4.YGrid = 'on';
app.UIAxes3_4.Position = [1 76 529 223];
% Create GuardarInformacindeGrficasPanel
app.GuardarInformacindeGrficasPanel = uipanel(app.ReconstruccionTab);
app.GuardarInformacindeGrficasPanel.Title = 'Guardar información de gráficas';
app.GuardarInformacindeGrficasPanel.Position = [8 12 517 53];
% Create GuardarButton
app.GuardarButton = uibutton(app.GuardarInformacindeGrficasPanel, 'push');
app.GuardarButton.ButtonPushedFcn = createCallbackFcn
(app, @GuardarButtonPushed, true);
```

```
app.GuardarButton.Position = [441 5 65 23];
app.GuardarButton.Text = 'Guardar';
% Create csvLabel
app.csvLabel = uilabel(app.GuardarInformacindeGrficasPanel);
app.csvLabel.Position = [391 5 26 22];
app.csvLabel.Text = '.csv';
% Create NombredearchivoEditFieldLabel
app.NombredearchivoEditFieldLabel = uilabel(app.GuardarInformacindeGrficasPanel);
app.NombredearchivoEditFieldLabel.HorizontalAlignment = 'right';
app.NombredearchivoEditFieldLabel.Position = [1 5 106 22];
app.NombredearchivoEditFieldLabel.Text = 'Nombre de archivo';
% Create Nombredearchivoaguardar
app.Nombredearchivoaguardar = uieditfield
(app.GuardarInformacindeGrficasPanel, 'text');
app.Nombredearchivoaguardar.ValueChangedFcn = createCallbackFcn
(app, @NombredearchivoaguardarValueChanged, true);
app.Nombredearchivoaguardar.Position = [122 5 270 22];
app.Nombredearchivoaguardar.Value = 'a1';
% Create ReconstruccinDCyestimacinenpixelesLabel
app.ReconstruccinDCyestimacinenpixelesLabel = uilabel(app.UIFigure);
app.ReconstruccinDCyestimacinenpixelesLabel.FontSize = 36;
app.ReconstruccinDCyestimacinenpixelesLabel.Position = [47 543 713 47];
app.ReconstruccinDCyestimacinenpixelesLabel.Text =
'Reconstrucción DC y estimación en píxeles';
% Create CargarsealparacalibrarMximosyMnimosPanel
app.CargarsealparacalibrarMximosyMnimosPanel = uipanel(app.UIFigure);
app.CargarsealparacalibrarMximosyMnimosPanel.Title =
'Cargar señal para calibrar (Máximos y Mínimos)';
app.CargarsealparacalibrarMximosyMnimosPanel.Position = [13 344 427 93];
% Create CargarButton
app.CargarButton = uibutton(app.CargarsealparacalibrarMximosyMnimosPanel, 'push');
app.CargarButton.ButtonPushedFcn = createCallbackFcn
(app, @CargarButtonPushed, true);
app.CargarButton.BackgroundColor = [1 1 1];
app.CargarButton.Position = [315 9 100 23];
app.CargarButton.Text = 'Cargar';
```

```
% Create ArchivoEncontrado
app.ArchivoEncontrado = uilabel(app.CargarsealparacalibrarMximosyMnimosPanel);
app.ArchivoEncontrado.BackgroundColor = [0.8118 0.9216 1];
app.ArchivoEncontrado.Position = [11 9 300 22];
app.ArchivoEncontrado.Text = '';
% Create BuscarArchivoButton
app.BuscarArchivoButton =
uibutton
(app.CargarsealparacalibrarMximosyMnimosPanel, 'push');
app.BuscarArchivoButton.ButtonPushedFcn = createCallbackFcn
(app, @BuscarArchivoButtonPushed, true);
app.BuscarArchivoButton.Position = [12 35 100 22];
app.BuscarArchivoButton.Text = 'Buscar Archivo';
% Create RecotarSealaRegiondeIntersyclcularVoffsetconlosprimerosPanel
app.RecotarSealaRegiondeIntersyclcularVoffsetconlosprimerosPanel =
uipanel(app.UIFigure);
app.RecotarSealaRegiondeIntersyclcularVoffsetconlosprimerosPanel.Title =
'Recotar señal a region de rnterés y cálcular voltaje
offset con los primeros N datos';
app.RecotarSealaRegiondeIntersyclcularVoffsetconlosprimerosPanel.Position =
[13 216 427 121];
% Create InicioSpinnerLabel
app.InicioSpinnerLabel = uilabel
(app.RecotarSealaRegiondeIntersyclcularVoffsetconlosprimerosPanel);
app.InicioSpinnerLabel.HorizontalAlignment = 'right';
app.InicioSpinnerLabel.Position = [31 77 33 22];
app.InicioSpinnerLabel.Text = 'Inicio';
% Create InicioSpinner
app.InicioSpinner = uispinner
(app.RecotarSealaRegiondeIntersyclcularVoffsetconlosprimerosPanel);
app.InicioSpinner.Limits = [1 19000];
app.InicioSpinner.ValueDisplayFormat = '%.0f';
app.InicioSpinner.ValueChangedFcn =
createCallbackFcn(app, @InicioSpinnerValueChanged, true);
app.InicioSpinner.Position = [12 56 71 22];
app.InicioSpinner.Value = 1;
```

```
% Create FinSpinnerLabel
app.FinSpinnerLabel = uilabel
(app.RecotarSealaRegiondeIntersyclcularVoffsetconlosprimerosPanel);
app.FinSpinnerLabel.HorizontalAlignment = 'right';
app.FinSpinnerLabel.Position = [120 77 25 22];
app.FinSpinnerLabel.Text = 'Fin';
% Create FinSpinner
app.FinSpinner = uispinner
(app.RecotarSealaRegiondeIntersyclcularVoffsetconlosprimerosPanel);
app.FinSpinner.Limits = [1000 20000];
app.FinSpinner.ValueDisplayFormat = '%.0f';
app.FinSpinner.ValueChangedFcn = createCallbackFcn
(app, @FinSpinnerValueChanged, true);
app.FinSpinner.Position = [98 56 70 22];
app.FinSpinner.Value = 2000;
% Create DatosoffsetLabel
app.DatosoffsetLabel = uilabel
(app.RecotarSealaRegiondeIntersyclcularVoffsetconlosprimerosPanel);
app.DatosoffsetLabel.Position = [309 72 112 30];
app.DatosoffsetLabel.Text = 'Datos offset (N)';
% Create DatosoffsetNSpinner
app.DatosoffsetNSpinner = uispinner
(app.RecotarSealaRegiondeIntersyclcularVoffsetconlosprimerosPanel);
app.DatosoffsetNSpinner.Limits = [0 1000];
app.DatosoffsetNSpinner.ValueChangedFcn = createCallbackFcn
(app, @DatosoffsetNSpinnerValueChanged, true);
app.DatosoffsetNSpinner.Position = [321 56 64 22];
% Create DatossuavizadokSpinnerLabel
app.DatossuavizadokSpinnerLabel = uilabel
(app.RecotarSealaRegiondeIntersyclcularVoffsetconlosprimerosPanel);
app.DatossuavizadokSpinnerLabel.Position = [176 77 111 22];
app.DatossuavizadokSpinnerLabel.Text = 'Media móvil (k)';
% Create DatossuavizadokSpinner
app.DatossuavizadokSpinner = uispinner
(app.RecotarSealaRegiondeIntersyclcularVoffsetconlosprimerosPanel);
app.DatossuavizadokSpinner.Limits = [3 100];
```

```
app.DatossuavizadokSpinner.ValueChangedFcn = createCallbackFcn
(app, @DatossuavizadokSpinnerValueChanged, true);
app.DatossuavizadokSpinner.Position = [199 56 65 22];
app.DatossuavizadokSpinner.Value = 25;
% Create RecortarysuavizarButton
app.RecortarysuavizarButton = uibutton
(app.RecotarSealaRegiondeIntersyclcularVoffsetconlosprimerosPanel, 'push');
app.RecortarysuavizarButton.ButtonPushedFcn= createCallbackFcn
(app, @RecortarysuavizarButtonPushed, true);

app.RecortarysuavizarButton.BackgroundColor = [1 1 1];
app.RecortarysuavizarButton.Position = [144 4 113 31];
app.RecortarysuavizarButton.Text = 'Recortar y suavizar';
% Create CondicinparadeteccinPanel
app.CondicinparadeteccinPanel = uipanel(app.UIFigure);
app.CondicinparadeteccinPanel.Title = 'Condición para detección';
app.CondicinparadeteccinPanel.Position = [13 86 427 123];
% Create ResultadoButton
app.ResultadoButton = uibutton(app.CondicinparadeteccinPanel, 'push');
app.ResultadoButton.ButtonPushedFcn = createCallbackFcn
(app, @ResultadoButtonPushed, true);
app.ResultadoButton.Position = [345 6 65 23];
app.ResultadoButton.Text = 'Resultado';
% Create UmbralSpinnerLabel
app.UmbralSpinnerLabel = uilabel(app.CondicinparadeteccinPanel);
app.UmbralSpinnerLabel.HorizontalAlignment = 'right';
app.UmbralSpinnerLabel.Position = [5 73 44 22];
app.UmbralSpinnerLabel.Text = 'Umbral';
% Create UmbralSpinner
app.UmbralSpinner = uispinner(app.CondicinparadeteccinPanel);
app.UmbralSpinner.Step = 0.01;
app.UmbralSpinner.Limits = [0 Inf];
app.UmbralSpinner.ValueDisplayFormat = '%5.5f';
app.UmbralSpinner.ValueChangedFcn = createCallbackFcn
(app, @UmbralSpinnerValueChanged, true);
app.UmbralSpinner.Position = [61 73 78 22];
```

```
% Create DistanciaentredatosparanuevadeteccinSpinnerLabel
app.DistanciaentredatosparanuevadeteccinSpinnerLabel = uilabel
(app.CondicinparadeteccinPanel);
app.DistanciaentredatosparanuevadeteccinSpinnerLabel.VerticalAlignment = 'top';
app.DistanciaentredatosparanuevadeteccinSpinnerLabel.WordWrap = 'on';
app.DistanciaentredatosparanuevadeteccinSpinnerLabel.Position = [188 63 129 32];
app.DistanciaentredatosparanuevadeteccinSpinnerLabel.Text =
'Distancia entre datos para nueva detección';
% Create DistanciaentredatosparanuevadeteccinSpinner
app.DistanciaentredatosparanuevadeteccinSpinner = uispinner
(app.CondicinparadeteccinPanel);
app.DistanciaentredatosparanuevadeteccinSpinner.Limits = [0 Inf];
app.DistanciaentredatosparanuevadeteccinSpinner.ValueDisplayFormat = '%.0f';
app.DistanciaentredatosparanuevadeteccinSpinner.ValueChangedFcn
= createCallbackFcn
(app, @DistanciaentredatosparanuevadeteccinSpinnerValueChanged, true)

app.DistanciaentredatosparanuevadeteccinSpinner.Position = [326 73 83 22];
% Create PromediarpicosmnimosdetectadosCheckBox
app.PromediarpicosmnimosdetectadosCheckBox = uicheckbox
(app.CondicinparadeteccinPanel);
app.PromediarpicosmnimosdetectadosCheckBox.Text =
'Promediar picos mínimos detectados?';
app.PromediarpicosmnimosdetectadosCheckBox.Position = [6 13 228 22];
% Create PromediosLabel
app.PromediosLabel = uilabel(app.UIFigure);
app.PromediosLabel.HorizontalAlignment = 'center';
app.PromediosLabel.FontWeight = 'bold';
app.PromediosLabel.Position = [234 22 67 22];
app.PromediosLabel.Text = 'Promedios';
% Create MinLabel
app.MinLabel = uilabel(app.UIFigure);
app.MinLabel.HorizontalAlignment = 'center';
app.MinLabel.Position = [293 59 25 22];
app.MinLabel.Text = 'Min';
% Create MaxLabel
```

```
app.MaxLabel = uilabel(app.UIFigure);
app.MaxLabel.HorizontalAlignment = 'center';
app.MaxLabel.Position = [216 59 28 22];
app.MaxLabel.Text = 'Max';
% Create MinResult
app.MinResult = uilabel(app.UIFigure);
app.MinResult.BackgroundColor = [1 1 0];
app.MinResult.HorizontalAlignment = 'center';
app.MinResult.Position = [276 43 60 22];
app.MinResult.Text = '.......';
% Create MaxResult
app.MaxResult = uilabel(app.UIFigure);
app.MaxResult.BackgroundColor = [1 1 0];
app.MaxResult.HorizontalAlignment = 'center';
app.MaxResult.Position = [202 43 56 22];
app.MaxResult.Text = '.......';
% Create LimpiarButton
app.LimpiarButton = uibutton(app.UIFigure, 'push');
app.LimpiarButton.ButtonPushedFcn = createCallbackFcn
(app, @LimpiarButtonPushed, true);
app.LimpiarButton.Position = [365 33 63 37];
app.LimpiarButton.Text = 'Limpiar';
% Create ProcesoButtonGroup
app.ProcesoButtonGroup = uibuttongroup(app.UIFigure);
app.ProcesoButtonGroup.SelectionChangedFcn = createCallbackFcn
(app, @ProcesoButtonGroupSelectionChanged, true);
app.ProcesoButtonGroup.Title = 'Proceso';
app.ProcesoButtonGroup.Position = [13 445 427 75];
% Create CalibracinMximosyMnimosButton
app.CalibracinMximosyMnimosButton = uiradiobutton(app.ProcesoButtonGroup);
app.CalibracinMximosyMnimosButton.Text = 'Calibración (Máximos y Mínimos)';
app.CalibracinMximosyMnimosButton.Position = [11 29 200 22];
app.CalibracinMximosyMnimosButton.Value = true;
% Create AproximacinDCenVypixelesButton
app.AproximacinDCenVypixelesButton = uiradiobutton(app.ProcesoButtonGroup);
app.AproximacinDCenVypixelesButton.Text = 'Aproximación DC en voltaje y píxeles';
```

```
app.AproximacinDCenVypixelesButton.Position = [11 7 194 22];
% Create DimensionesdelmonitorLabel
app.DimensionesdelmonitorLabel = uilabel(app.UIFigure);
app.DimensionesdelmonitorLabel.FontSize = 14;
app.DimensionesdelmonitorLabel.FontWeight = 'bold';
app.DimensionesdelmonitorLabel.Position = [782 569 172 22];
app.DimensionesdelmonitorLabel.Text = 'Dimensiones del monitor';
% Create valorAlto
app.valorAlto = uilabel(app.UIFigure);
app.valorAlto.BackgroundColor = [1 1 0];
app.valorAlto.Position = [807 547 63 22];
app.valorAlto.Text = '.......';
% Create valorAncho
app.valorAncho = uilabel(app.UIFigure);
app.valorAncho.BackgroundColor = [1 1 0];
app.valorAncho.Position = [925 547 72 22];
app.valorAncho.Text = '.......';
% Create AltoLabel
app.AltoLabel = uilabel(app.UIFigure);
app.AltoLabel.Position = [782 547 26 22];
app.AltoLabel.Text = 'Alto';
% Create AnchoLabel
app.AnchoLabel = uilabel(app.UIFigure);
app.AnchoLabel.Position = [887 548 39 22];
app.AnchoLabel.Text = 'Ancho';
% Create DatosparacalibracionobtenidosLampLabel
app.DatosparacalibracionobtenidosLampLabel = uilabel(app.UIFigure);
app.DatosparacalibracionobtenidosLampLabel.WordWrap = 'on';
app.DatosparacalibracionobtenidosLampLabel.Position = [27 38 120 33];
app.DatosparacalibracionobtenidosLampLabel.Text =
'Datos para calibración obtenidos';
% Create DatosparacalibracionobtenidosLamp
app.DatosparacalibracionobtenidosLamp = uilamp(app.UIFigure);
app.DatosparacalibracionobtenidosLamp.Position = [156 41 27 27];
app.DatosparacalibracionobtenidosLamp.Color = [0.149 0.149 0.149];
% Show the figure after all components are created
```

```
app.UIFigure.Visible = 'on';
end
end
% App creation and deletion
methods (Access = public)
% Construct app
function app = mainx
% Create UIFigure and components
createComponents(app)
% Register the app with App Designer
registerApp(app, app.UIFigure)
% Execute the startup function
runStartupFcn(app, @startupFcn)
if nargout == 0
clear app
end
end
% Code that executes before app deletion
function delete(app)
% Delete UIFigure when app is deleted
delete(app.UIFigure)
end
end
end
```